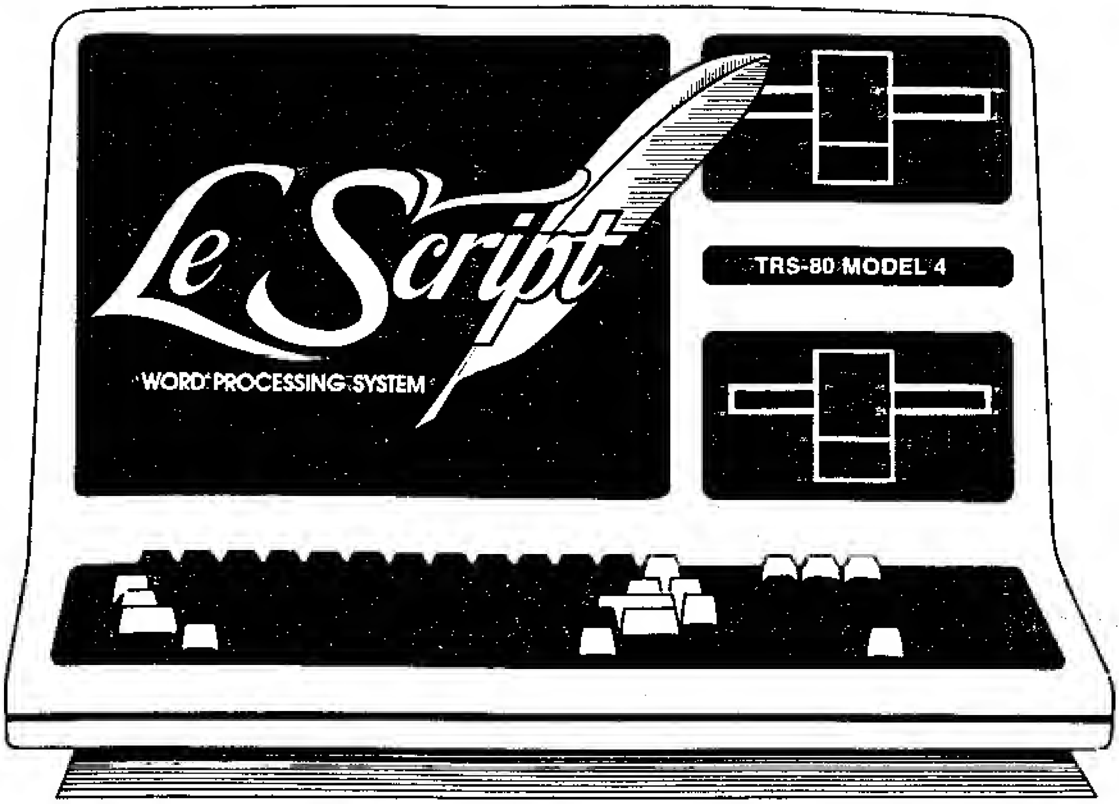


"World Class"

WORD PROCESSING SYSTEM

For TRS-80 Models I, III, IV, Lobo MAX-80, LNW, PMC-81

Version 1.1

Model 1, LNW, and PMC-81

Manual B

ANITEK SOFTWARE PRODUCTS
P.O. BOX 361136
MELBOURNE FL 32936
1-305-259-9397

LIMITED GUARANTEE

This product is sold on an "AS IS" basis and is without warranty. The advertisements used to promote this product and the manual provided with this product are believed to be fair descriptions of this product's capabilities but are in no way guarantees of actual performance. If the purchaser discovers upon receiving this product that the disk media is unreadable or damaged in such a way that it is unusable, he may send the disk back to ANITEK for exchange within 10 days of the date of purchase. Returns for exchange must include a photocopy of the original sales slip in order to receive the exchange.

WAIVER OF LIABILITY

ANITEK shall have no liability or responsibility to purchaser or any other person or entity regarding liability, loss, or damage caused, or alleged to be caused, directly or indirectly, by use of programs, program manuals, or segments of either which are part of this or any other product provided by ANITEK. This includes, but is not limited to, any loss of business or anticipated profits resulting from such loss of business, or any interruption of service or consequential damages, whether physical, monetary, material, or of any other kind, resulting from any use of such programs or program manuals, or segments of either.

DECLARATION OF COPYRIGHT

This product, both manual and operating program, is copyrighted. ANITEK authorizes each LeScript purchaser the right to make back-up copies for the sole personal use of said purchaser. Any other duplication of the LeScript manual and/or operating program, in whole or in part, is unlawful and strictly prohibited.

You probably realize as well as we do that we could have made this program copy-protected to insure that unauthorized copies could never be made. Most of our competition does just that, and that is their prerogative. However, we feel that this would have greatly limited the program's usefulness to those people who paid for the program and should be entitled to full copy privileges including transferring it to a more convenient media such as a hard disk drive if the purchaser so desired. Please don't abuse this privilege. Make as many copies as you need but none for anyone else. In so doing, you will be protecting your own interests in this product's market success, which helps you because it gives us greater opportunities to continue to develop, support, and improve LeScript.

REQUIRED HARDWARE

1. TRS-80 Model I, III, 4, Lobo Max-80, LNW, or PMC-81.

2. 48K or more memory.

3. 1 disk drive minimum required to run LeScript. (LeScript will have to be copied to a DOS disk, which may require 2 drives depending on the DOS used.)

4. One of the following DOS's: TRSDOS, LDOS, DOSPLUS, NEWDOS, NEWDOS/80, or MULTIDOS (single or double density). Model 4 owners may run LeScript under any Model III DOS or under TRSDOS 6.0.

5. Lower case modification (Model 1 only).

6. Line printer (can be either parallel or serial).

Note: Any standard line printer will work. However, using one which LeScript specially supports (see section 5.1) will give you more word processing power than using another brand.

Note: Serial port on MAX-80 will not operate under LeScript.

Note: The Model 4 and the Max-80 will have a 80 X 24 display. The Model I, III, LNW, and PMC-81 will have a 64 X 16 display.

RECOGNITION OF TRADEMARKS

1. TRS-80, TRSDOS, Line Printer IV, Line Printer VI, Line Printer VIII, DMP-100, DMP-200, DMP-400, DMP-500, DMP-2100, DWP-410, Daisy Wheel II, Scripsit, and Radio Shack are trademarks of Tandy Corporation.

2. MAX-80 is a trademark of Lobo Systems.

3. LNW is a trademark of LNW Research Corporation.

4. PMC-81 is a trademark of Personal Micro Computers.

5. Microline is a trademark of OKI Electric Industry Company, LTD.

6. Starwriter F-10, Starwriter FP-1500, and Prowriter 8510 are trademarks of C. Itoh Electronics, Inc.

7. Epson, MX-80, MX-100, RX-80, FX-80, FX-100, Graftrax, and Graftrax Plus are trademarks of Epson America, Inc.

8. Gemini-10 and Gemini-15 are trademarks of Star Micronics Corporation.

9. Microprism 480 is a trademark of IDS Corporation.

10. Centronics 737 and Centronics 739 are trademarks of Centronics Data Corporation.

11. Qume Sprint is a trademark of Qume Corporation.

12. Diablo 630 is a trademark of Diablo Systems, Inc.

13. Spinwriter and PC-8023A-C are trademarks of NEC America, Inc.

14. TEC 8500R, F-10, and DMP-85 are trademarks of Tec Corporation.

15. Brother HR-1 is a trademark of Brother Industries, LTD.

16. ComRiter CR-1 is a trademark of Comrex International, Inc.

17. Smith-Corona TP-1 is a trademark of Smith-Corona.

18. SILVER-REED is a trademark of Silver-Reed America, Inc.

19. LDOS is a trademark of Logical Systems, Inc.

20. NEWDOS and NEWDOS/80 are trademarks of Apparat, Inc.

21. DOSPLUS is a trademark of Micro Systems Software, Inc.

22. MULTIDOS is a trademark of Cosmopolitan Electronics Corporation.

23. ELECTRIC WEBSTER is a trademark of Cornucopia Software.

[illegible]

TABLE OF CONTENTS

(cont.)

1. GETTING STARTED

1.1. Fill out and send in the enclosed registration card so we can put you on our LeScript mailing list. If a card was not provided with your system, please write your name, of purchase on a sheet of paper and send it to the address on the cover of the manual.

1.2. Read the manual <u>cover to cover</u> before you do anything else. A thorough understanding of its contents is important. Pay special attention to Appendix C "ANSWERS TO COMMONLY ASKED QUESTIONS." Any questions you might still have at that point will probably be answered in that section. Once the manual is read and understood, remove the summary card from the back of this manual and refer to it as you use LeScript to help you remember what you read in the manual.

1.3. Copy LeScript and the example text files from the master disk to your DOS system disk using the procedure on the following page. The LeScript master disk contains no system software and, thus, will not operate by itself in drive 0. The file names you need to copy are LESCRIPT/CMD, LESCRIPT/DVR, SAMPLE/TXT, FORM/TXT, FORM/DAT. Some DOS's require 2 disk drives to make this copy. If you have any trouble with this part, refer to the section in your DOS operator's manual that discusses copying files from a TRSDOS Model I or Model III disk to your DOS Once the files from the original are copied, store the master disk away in a safe place; and don't ever use it for anything except making more copies.

1.4. Boot up your system with a DOS disk that contains LESCRIPT/CMD. Type "LESCRIPT" (no quotes), then hit the ENTER key. After a few seconds, the screen will blank except for three status lines at the top of the screen (explained in Section 3) and a blinking rectangle at the start of the next line down (this is the cursor, the point at which your typing takes place). LeScript is now running.

1.5. Now you can begin typing a text file. Move your cursor up to the name field with CLEAR-=. Type a name for your file. Move the cursor back down to the third line using the down arrow. Type a printer command line starting with a printer command character (CLEAR-;) and ending with a return character (ENTER). This first command line will contain your main formatting commands such as your printer type (5.1), tab positions (5.4), margins (5.9), etc. Page numbering printer command lines should follow right after this. (Review sections 5.23-27 and also see how it's done in the SAMPLE/TXT file).

1.6. Move your cursor down to the next line and type, just like you would on a typewriter. The word wrap makes sure words don't split between lines. The ENTER key is used to terminate paragraphs and to create blank lines. The tab key is CLEAR-K for paragraph indenting. CLEAR-L will delete a line. CLEAR-→ will delete characters. CLEAR-I will put you into insert mode and hit once again will take you out. Bolding, underlining, italics, condensed print, expanded print and many more text-enhancing commands are explained in sections 4 and 5 of this manual.

1.7. Once the text is finished, you may view it in near final form with CLEAR-V. In the view mode, the down arrow scrolls the text; and CLEAR-Z will get you back to text editing. CLEAR-P will cause the text to be printed. CLEAR-Z can be used to abort printing too. When done, store the text on the disk with CLEAR-F, and clear the screen using CLEAR-0.

1.8. The directory can be viewed with CLEAR-D and then the number key of the desired drive. Paging through the directory is done with the space bar. Files can be loaded into memory from the directory with CLEAR-G. And Files can be killed from the directory using CLEAR-K. CLEAR-Z will get you out of the directory and redisplay the text.

1.9. At this point, we suggest you take a look at and try printing the sample text files which were supplied on the LeScript master disk. SAMPLE/TXT will demonstrate many of the PRINT FORMAT commands and how they work (explained in section 5). FORM/TXT and FORM/DAT together give an example of how a form letter text file and form letter data file might be set up (explained in section 6).

1.10. Once you understand the manual and have tried some of the examples, we don't expect you to have any difficulties using LeScript. Even though LeScript is an extremely powerful text-processing tool, a lot of care and thought has gone into making it as easy to use as possible. We also give a free 30-day support period to each new LeScript owner, should you need extra assistance. Call with any questions you might have. (We don't accept collect calls.)

However, before spending your money on a phone call or taking the time to write (write if you're past 30 days), you might want to try rereading the sections in your DOS manual or the LeScript manual that pertain to your question. Also, refer back to the examples supplied on the disk to see how we did some of these things. Then look in Appendix C to see if we have given an answer to your question there. Just about everything one would ever need to know about LeScript is already found in this manual. You paid for it, <u>use it</u>.

PROCEDURE
to COPY from LeScript master disk to other DOS's on the Model 3

TRSDOS: Put the TRSDOS disk in drive 0 and the LeScript master in drive 1, boot up, type:

```
COPY LESCRIPT/CMD:1 :0
COPY LESCRIPT/DVR:1 :0
COPY SAMPLE/TXT:1 :0
COPY FORM/TXT:1 :0
COPY FORM/DAT:1 :0
```

MULTIDOS: Put the MULTIDOS disk in drive 0 and the LeScript master in drive 1, boot up, type:

```
COPY LESCRIPT/CMD:1 TO LESCRIPT/CMD:0
COPY LESCRIPT/DVR:1 TO LESCRIPT/DVR:0
COPY SAMPLE/TXT:1 TO SAMPLE/TXT:0
COPY FORM/TXT:1 TO FORM/TXT:0
COPY FORM/DAT:1 TO FORM/DAT:0
```

NEWDOS/80: Put the NEWDOS/80 disk in drive 0 and the LeScript master in drive 1, boot up, type:

```
COPY LESCRIPT/CMD:1 :0 SPDN=4
COPY LESCRIPT/DVR:1 :0 SPDN=4
COPY SAMPLE/TXT:1 :0 SPDN=4
COPY FORM/TXT:1 :0 SPDN=4
COPY FORM/DAT:1 :0 SPDN=4
```

DOSPLUS: Put the DOSPLUS disk in drive 0 and the LeScript master in drive 1, boot up, type:

```
CONVERT :1 :0 (V13)
```

LDOS: Put the LDOS disk in drive 0 and the LeScript master in drive 1, boot up, type:

```
CONV :1 :0      (answer all prompts with: Y)
```

PROCEDURE
to COPY from LeScript master disk
to other DOS's on the Model 1, LNW, or PMC-81

ALL DOS's: Put the DOS disk in drive 0 and the LeScript master in drive 1, boot up, type:

```
COPY LESCRIPT/CMD:1 TO LESCRIPT/CMD:0
COPY LESCRIPT/DVR:1 TO LESCRIPT/DVR:0
COPY SAMPLE/TXT:1 TO SAMPLE/TXT:0
COPY FORM/TXT:1 TO FORM/TXT:0
COPY FORM/DAT:1 TO FORM/DAT:0
```

PROCEDURE
to COPY from LeScript master disk
to other Model 4 TRSDOS 6.0

TRSDOS 6.0: Put the TRSDOS 6.0 disk in drive 0 and the LeScript master in drive 1, boot up, type:

```
CONV :1 :0      (answer all prompts with: Y)
```

PROCEDURE
to COPY from LeScript master disk
to Lobo MAX-80 LDOS

MAX-80 LDOS: Put the LDOS disk in drive 0 and the LeScript master in drive 1, boot up, type:

```
CONV :1 :0      (answer all prompts with: Y)
```

2. WORD-WRAP AND JUSTIFICATION

The operator of LeScript is free to type continuously, never worrying about where to break the lines or how the spacing between the words should be to achieve proper line justification. LeScript does this all for you automatically, using its advanced WORD-WRAP and LINE JUSTIFICATION capabilities which match and even surpass those found on the higher-priced word processing systems. Both of these features will become very powerful text editing tools for you once they are understood.

2.1. WORD-WRAP

If a word that you are typing will not fit on the line without spilling over the end of the line (see 3.2 for line length specifications), then LeScript will automatically take that word off that line and put it on the next line. At the same time, the line you were typing on automatically justifies on the screen.

Another part of the WORD-WRAP capabilities of LeScript is REVERSE WORD-WRAP. Any time you type a space on a line, LeScript will check to see if the word(s) from the beginning of the line to that space can fit on the previous line. If it can, then it is "wrapped backwards", so to speak, and put at the end of the previous line. The previous line is then re-justified, and the cursor is moved back to the second column of the current line (second, because the space is in the first). LeScript will continue to move words up to the end of the previous line with each space you type until the previous line is full.

While scrolling up, the WORD-WRAP function in LeScript will always put as many words on a line as will fit. Lines containing a RETURN character (▪)(4.47), however, are the exception. No text will ever be put on a line following a RETURN character while scrolling.

2.2. LINE JUSTIFICATION

LeScript has very advanced LINE JUSTIFICATION routines which automatically justify the lines of your text not only as they are printed, giving you a very professional-looking copy, but also right on the screen as you type, giving you the opportunity to see how the text is going to look before you ever print it. You can format your text using one (or a combination) of four different LINE JUSTIFICATION modes. They are:

JUSTIFY LEFT	All text lines start in the first character column to form a smooth margin on the left side of the screen and paper.
JUSTIFY RIGHT	All text lines end in the last character column to form a smooth margin on the right side of the screen and paper.
JUSTIFY BOTH	All text lines start and end in the first and last character columns, respectively, to form smooth margins on both sides of the screen and paper. The spacing between words is done automatically.
JUSTIFY CENTER	All text lines are centered on screen and on paper.

If desired, you can change the LINE JUSTIFICATION mode at any place in your text--having one paragraph set to JUSTIFY LEFT, the next to JUSTIFY CENTER, etc. The default justification mode is JUSTIFY LEFT. Refer to section 5.11 for instructions on how to set the LINE JUSTIFICATION mode within your text.

LeScript can take advantage of the full power of the proportional-space, text-printing feature of over 60 printers (see 5.1 for list). On these printers each line will be justified, as with any printer, but all the between-word spacings will be made exactly equal, and the spaces between characters will be uniformly increased to avoid excessive between-word spacing whenever necessary. The result is a printed page that looks almost like it was run off on an expensive typesetter.

Using LeScript with any one of these types of printers, you can also specify any character print density from 6.0 to 20.0 characters per line inch. The maximum print density you will actually be able to achieve and still have lines correctly justify will depend on the average character width for your printer. About 14 characters per inch is as tight as you will normally be able to print on a dot matrix printer and about 11 characters per inch on a daisy wheel. This manual was printed in proportional-space mode at a character density of 11.0 characters per inch using NEC PC-8023A-C printer. Refer to section 5.14 for instructions on how to specify CHARACTER DENSITY within your text.

3. STATUS LINES

The top three display lines on your screen are reserved for status information. These lines can never be scrolled off or removed from the screen.

When LeScript first comes up, the cursor will be positioned over to the left just under the third STATUS LINE. The only way to get the cursor into the STATUS LINES from this point (or from any point in the text portion of the display) is to hold down the CLEAR key and strike the "=" key. This is denoted CLEAR-= (see 4.50).

You will notice that LeScript will only allow the cursor to go into certain areas of the STATUS LINES and not others. This is a protective measure to keep you from typing over any part of the STATUS LINES that should not be changed. Also, some functions are not operable while the cursor is in the STATUS LINE (see beginning of section 4 for list). This, too, is precautionary to protect the STATUS LINES.

There are five fields in the first STATUS LINE. Two of them, the NAME FIELD and the WIDTH FIELD, are for you to type information in to tell LeScript certain things about the text you are working on. The other three, the WORD COUNT FIELD, the LINE COUNT FIELD, and the FREE MEMORY FIELD are for LeScript to write information in to tell you certain things about the text.

The second STATUS LINE contains the SEARCH FIELD and the REPLACE FIELD, which you use to type the SEARCH and REPLACE strings in while using those respective functions (4.37, 4.38).

The third STATUS LINE displays the COLUMN NUMBER GRID. This grid spans the entire width of the active text and will move horizontally as the text is moved horizontally. The grid numbers off the columns kind of like a yardstick, helping you judge where things are in your text relative to the left edge. The COLUMN GRID also uses colons (:) to show you where the tab stops are set. (See section 5.4 for more information on setting tab stops). When needed, LeScript will use the third STATUS LINE for displaying the STATUS MESSAGES. Each of these fields is discussed in detail on the following pages.

3.1. NAME FIELD

This is the field where you type the name of the text file. This is the name that LeScript will use when you go to file your text onto the disk. The name must follow the file specification format as dictated by your disk operating system (consult your DOS operator's manual) and must be no longer than 14 characters, including the extension and drive specification (if used). Passwords are not allowed as part of the file name.

Also, anytime you get a text file from the disk, unless there is already a name in the NAME FIELD at the time, LeScript will take the name that you have used in specifying that file and write it into the NAME FIELD for you. This saves you some trouble and it also helps to prevent you from writing the file back out to disk under the wrong name.

3.2. WIDTH FIELD

This is the field where you type the maximum width (in characters) of your copy. This is also referred to as maximum line length in other parts of this manual. Any width from 5 to 252 may be entered in this field. Portions of text can be made narrower than this maximum specification by implementing the TEXT INDENTING printer commands as described in section 5.12.

Horizontal scrolling becomes active any time you assign a text width that is greater than the width of your display. This means that if you have 100 typed into the WIDTH FIELD, for example, your display screen will act as a "window" that can be moved (using the left or right arrow keys) back and forth horizontally to view the full width of your 100 column text.

It is possible to type your text in one width and then change it later to a different width if you so desire. Just type the new width in the WIDTH FIELD and hit CLEAR-T (4.39). Voila, the text is automatically reformatted, word-wrapped, and line justified to the new line length.

You may also be happy to know that the information in the WIDTH FIELD is saved as part of the file data every time you file the text (or even a portion of the text--see 4.21) onto disk. Then, the next time you call that text file back to the screen to work on it, LeScript will automatically extract that information from the file and write it into the WIDTH FIELD for you, saving you a lot of time and trouble. If, however, you are getting a text file from disk to insert it into text that is already in memory, the width data of that second file is just stripped from the text data of the file and ignored, maintaining the width of the first file.

The WIDTH FIELD is used in a slightly different way when editing files other than LeScript-type text files. During these times, the width field will contain a three-character abbreviation for the type of file being edited. If you are using LeScript to create one of these non-LeScript-standard files, you will need to type the appropriate three characters in the WIDTH FIELD before you start. If you are using LeScript to edit a pre-existing file, then the correct three-character abbreviation will appear in the WIDTH FIELD automatically as the file is being loaded into memory from the disk. The various types of non-standard files that LeScript can create and/or edit, along with their three-character abbreviations and explanations of the unique characteristics of each type of file, are given in Appendix A in the back of this manual. If you have any desire to use LeScript for editing BASIC programs, Assembly language source files, or other word processing Assembly language source files, or other word processing systems' text files, it is recommended that you review Appendix A.

3.3. WORD COUNT FIELD

This field contains a dynamic count of the total number of words in your text file, both on the screen and in memory, and is updated each time you strike a key. A word is defined by a character or a string of characters separated from other characters by one or more SPACE characters ( ), BLANK characters (▒), END-OF-SENTENCE characters (■), TABs, UNDERLINED SPACE characters (_), the start of a line, or the end of a line. Characters in printer command lines (lines starting with PRINTER COMMAND characters; see 4.16) are not counted. The information contained in this field is maintained by LeScript and cannot be changed manually by the user.

3.4. LINE COUNT FIELD

This field contains a dynamic count of the total number of lines in your text file, both on the screen and in memory, and is updated each time you strike a key. A text line is equal to one display line of your text. Printer command lines (lines starting with PRINTER COMMAND characters; see 4.16) are not counted. The information contained in this field is maintained by LeScript and cannot be changed manually by the user.

3.5. FREE MEMORY COUNT FIELD

This field contains a dynamic count of the unused memory space still available for character entry, and is updated each time you strike a key. The information contained in this field is maintained by LeScript and cannot be changed manually by the user.

3.6. SEARCH FIELD

If there is some string of characters that you want to SEARCH through your text for, you can have LeScript do it for you automatically. Type the string of characters in the SEARCH FIELD (first field of the second status line--maximum 28 characters) and hit CLEAR-S (4.37). LeScript will automatically scan the entire text, from the cursor down, looking for that string of characters. If found, the cursor will stop on the first character of the string. If not found, the cursor will stop on the next empty line after the end of the text.

When entering the character string in the SEARCH FIELD, you can use upper or lower case; it really doesn't matter. LeScript will look for that string in any combination of upper or lower case.

The BLANK character (▒) in the SEARCH string is used as a wild card. Use it when you wish to SEARCH for a string in which some of the characters in the string have to be certain things but other characters can be anything.

Use the RETURN character (▪) in the SEARCH FIELD to define the end of a SEARCH string; otherwise it is assumed to be the right most non-space character in the field. This is necessary if you want to search for the word "at" but you don't want the cursor to stop on the word "attack". In this case, the SEARCH string should be entered: space, "a", "t", space, RETURN character. The space after the "at" is now taken to be the last character of the string, and LeScript will know just to stop on the word "at" and not "atom" or "cat".

3.7. REPLACE FIELD

Sometimes when you SEARCH for a string of characters you will want to REPLACE it with another string of characters. You can do so using this field and the REPLACE function (4.38).

Type the replacement string of characters in the REPLACE FIELD (second field of the second status line) just as you would want it to appear in your text. BLANK characters (▒) count as actual BLANK characters. The case of each character (upper or lower) does matter, so type each character in the case that you want it. The RETURN character (▪) can be used in the REPLACE FIELD the same way it is used in the SEARCH FIELD if you want the REPLACE string to end with a space instead of a character.

Once you have typed the replacement string in the REPLACE FIELD, hit CLEAR-S to SEARCH for the text string that you are going to REPLACE, then hit CLEAR-R. The string of characters that you have SEARCHED for will be removed at the cursor, and a copy of the string of characters in the REPLACE FIELD will be automatically written in its place.

The REPLACE string, by definition, is a replacement for the SEARCH string. Thus, if CLEAR-R is hit at any time that the cursor is not positioned at the start of a portion of text that matches the SEARCH string, nothing will happen.

While we are on the subject, you might as well be introduced to three more capabilities which you can achieve on LeScript playing various tricks with the SEARCH and REPLACE fields. These are AUTOMATIC SEARCH AND REPLACE, AUTOMATIC SEARCH AND DELETE, and AUTOMATIC TEXT INSERTION.

AUTOMATIC SEARCH AND REPLACE is a function that will automatically search through your entire text (from the cursor down) and change every occurrence of the SEARCH string with a copy of the REPLACE string. This is done by making the appropriate entries in the SEARCH and REPLACE fields and hitting CLEAR-A (4.18).

AUTOMATIC SEARCH AND DELETE will automatically search through the entire text file (from the cursor down) and delete every occurrence of the SEARCH string that it finds. This is done by typing the appropriate entry in the SEARCH field, leaving the REPLACE field blank, and hitting CLEAR-A.

AUTOMATIC TEXT INSERTION will cause a string of text up to 28 characters in length to be inserted at the cursor with a single key-stroke. This is done by typing the appropriate string of characters in the REPLACE field, moving the cursor to the desired location(s) in your text, and hitting CLEAR-R.

3.8. COLUMN GRID LINE

The third STATUS line is the COLUMN GRID LINE. This line numbers off the character columns by 10's (10,20,30,etc.). The COLUMN GRID LINE will also be the same length as the text lines as set in the width field (3.2) and will always stay "glued" to the text. That is to say, if you move the text horizontally left or right (called horizontal scrolling), the COLUMN GRID LINE will move with it. This is to enable you to judge accurately how far any text item is from the left margin when the left margin is out of view.

You will also notice that there are colon characters (:) in the COLUMN GRID LINE. These characters show you where your tab stops are set. The places they are set at initially are the default tab stops for LeScript. You can vary and adjust these to suit your own preference by using the TABS printer command (5.4).

3.9. STATUS MESSAGE FIELD

You cannot see this field when LeScript first comes up, but it really is there. Actually, under certain status conditions, the COLUMN GRID LINE is temporarily removed. This leaves the entire third line available for the STATUS MESSAGE. The appropriate STATUS MESSAGE flashes on the screen a few times, letting you know what has happened. When this happens, you should take whatever measures are necessary to correct the condition, such as delete an obsolete file if your disk is full, etc. As soon as a key is hit, the message goes away; and the COLUMN GRID LINE returns.

The ERROR MESSAGES that may display and the section of the manual that apply are as follows:

DISK ERROR, TRY AGAIN	4.23, 4.24, 4.32
DISK FULL	" "
WRITE PROTECTED DISKETTE	" "
DISK NOT READY	" "
FILE PROTECTED	" "
NO SUCH FILE	" "
DIRECTORY FULL	" "
MEMORY FULL	what it says.
BAD BLOCK	4.20, 4.31, 4.32
QUEUE IS FULL	4.35
ONCE MORE TO CLEAR	4.51
ONCE MORE TO DELETE	4.46
ONCE MORE TO KILL	4.28
ONCE MORE TO EXIT TO DOS	4.52
LOADING PRINTER DRIVER	4.34
UNABLE TO LOAD PRINTER DRIVER	4.34
ZORLOF FILE - CHECK PRINTER COMMANDS	4.24
HIT SPACE BAR TO CONTINUE	4.34, 5.5

4. EDITING FUNCTIONS

Editing functions are those which assist the operator in preparing and manipulating text while it's still on the screen
preparing and manipulating text while it's still on the screen
and in the computer's memory. These are functions such as: moving a block of text from one place to another, deleting, searching for a particular word(s), scrolling, or calling up a disk directory, just to name a few.

The following pages contain summaries of all 62 editing functions available to the LeScript user, how each is invoked, and an explanation of how each works. Most functions are invoked by holding down the CLEAR key and then striking one of the other keys. The SEARCH function is one example. It is denoted in this section like:

CLEAR-S

Others are invoked by holding the SHIFT key down and striking one of the other keys. For example, SCROLL UP will be denoted:

SHIFT-↑

Others are invoked by holding down both the SHIFT key and the CLEAR key and hitting one of the other keys. For example, the LEFT BRACKET character ([) is entered with:

SHIFT-CLEAR-(

And still others like CURSOR RIGHT are invoked by a single key:

→

Note: On those computers that have it, the CONTROL key may be used instead of the CLEAR key.

Note: The following functions are not operative while the cursor is in either one of the STATUS LINES:

4.11	DELETE CHARACTER
4.12	DELETE CHARACTER BACKWARDS
4.20	COPY BLOCK
4.24	GET TEXT FROM DISK
4.29	TAB
4.30	DELETE LINE
4.31	MOVE BLOCK
4.33	OPEN LINE
4.42	DELETE WORD
4.44	SPLIT TEXT
4.46	DELETE BLOCK

4.1. CURSOR RIGHT →

The cursor is moved one column to the right. If the cursor is at the right edge of the display screen, the text will move to the left. After the cursor has reached the last column of the line, the cursor will then skip down to the first column of the next line if your text is not in a horizontally scrolled position. Otherwise, the cursor will stop in the last column of the line.

4.2. CURSOR LEFT ←

The cursor is moved one column to the left. If the cursor is at the left edge of your display screen, the text will move to the right instead. No action is taken if the cursor is in the first column of the line.

4.3. CURSOR UP ↑

The cursor is moved one line up. If the cursor is on the top line of the display, the text will be scrolled down instead.

4.4. CURSOR DOWN ↓

The cursor is moved one line down. If the cursor is on the bottom text line of the display, the text will be scrolled up instead.

4.5. CURSOR FAR RIGHT SHIFT-→

The cursor is moved to one column past the last character of the line. This function can also be used to place the cursor in the right most field of the STATUS LINES.

4.6. CURSOR FAR LEFT SHIFT-←

The cursor is moved to the first column of the line. This function can also be used to place the cursor in the left most field of the STATUS LINES.

4.7. SCROLL UP SHIFT-↑

The text is scrolled up one line.

4.8. SCROLL DOWN SHIFT-↓

The text is scrolled down one line.

4.9. PAGE UP CLEAR-↑

The text is scrolled up 13 lines if the maximum text width as set in the WIDTH FIELD is 64 characters or less, 6 lines if greater than 64 (see 3.2).

4.10. PAGE DOWN CLEAR-↓

Same as 4.9 except down instead of up.

4.11. DELETE CHARACTER CLEAR-→

The character at the cursor is deleted, and the text to the right of the cursor is shifted left one column to fill the gap. If the cursor is to the right of the last character on the line, then the cursor will move to the first column of the next line. If the cursor is in the first column of a blank line, then all the text below the cursor will scroll up one line to fill the gap.

Note: When the DELETE CHARACTER function is used on an expanded character, both the character and the vertical line in front of it (¦)(used to indicate that it's an expanded character, see 4.43) are deleted.

4.12. DELETE CHARACTER BACKWARDS CLEAR-←

Same as 4.11 except that the cursor is moved one column to the left first. No action is taken if the cursor is in the first column.

4.13. LOWER CASE CLEAR-.

An alpha character (A-Z) at cursor is changed to lower case (if not already), and the cursor is moved one column to the right.

4.14. UPPER CASE CLEAR-,

An alpha character (A-Z) at cursor is changed to upper case (if not already), and the cursor is moved one column to the right.

4.15. CAPITAL LOCK — CLEAR-/

All alpha characters (A-Z) entered after hitting CLEAR-/ will be in upper case. Hit CLEAR-/ a second time to terminate.

4.16. PRINTER COMMAND CHARACTER — CLEAR-;

A PRINTER COMMAND character (▮) is entered at the cursor. All characters on the same line and following this character will be assumed by LeScript to be printer commands. Also, all lines starting with this character will be temporarily stripped from the text while PRINTING (4.34) or VIEWING (4.41).

Note: If the PRINTER COMMAND character is typed at some place other than the first character of a line, LeScript will automatically put it, and the characters which follow, on their own line as soon as that line is scrolled off and then scrolled back onto the screen or when CLEAR-J is struck (4.27).

4.17. PRINT SCREEN — CLEAR-@

When CLEAR-@ is struck, the entire text file that is in memory at the time will get sent to the printer in the form in which it appears on the display screen, not in its final form. If CLEAR-@ is struck while the directory is on the display screen (4.21), a dump of the directory for that disk will get sent to the printer. The type of printer assumed (parallel, serial, baud rate, etc.) will be that which was last specified when printing a text file (see 5.1, 5.2, and 5.3). If there was none previously specified, then the default is the parallel line printer. The function will terminate automatically when the file (or directory) is fully printed, or manually by hitting CLEAR-Z.

4.18. AUTOMATIC SEARCH AND REPLACE — CLEAR-A

All occurrences of the SEARCH string in the text from the cursor down are replaced by the REPLACE string. Terminate the function by hitting CLEAR-Z. See 3.6 and 3.7 for more information on how the SEARCH, REPLACE, AUTOMATIC SEARCH AND DELETE functions work.

4.19. BLOCK MARKER INSERT CLEAR-B

A BLOCK MARKER character (■) is inserted into the text just ahead of the cursor. Two such BLOCK MARKER characters must be used to define a block of text, one for its beginning and one for its end, before the COPY BLOCK (4.20), MOVE BLOCK (4.31), or STORE BLOCK (4.33) functions can be implemented. Only one BLOCK MARKER character is needed when implementing the DELETE BLOCK function (4.46).

4.20. COPY BLOCK CLEAR-C

Once a block of text has been defined (4.19), that block can be copied to any place in your text simply by moving the cursor to that place and hitting CLEAR-C. The original block and the block markers will remain where they were. The command can be immediately invoked again, or the block markers can be removed by hitting CLEAR-Q (see 4.36)

4.21. DIRECTORY CLEAR-D

At any time, whether text is on the screen or not, you can get a DIRECTORY of any disk by hitting CLEAR-D. The DIRECTORY will give you a complete list of files found on any disk on your system, the length of each file in bytes and granules, the amount of free space of each disk in granules, and the PRINT CHAIN list (4.35). Upon hitting CLEAR-D the first directory to be displayed is the one for disk drive 0. The directory for any other drive can be examined just by hitting 0, 1, 2, 3, etc. on the keyboard for the desired drive.

Only one screen-full of file names can appear on your display at one time. By hitting the space bar on your keyboard, the next set of file names from disk can be examined. The same PRINT CHAIN list appears on the right side of the screen no matter what directory page is displayed on the left side of the screen. To exit from the DIRECTORY function, hit CLEAR-Z and your text will return to the screen.

Hitting CLEAR-@ while the directory is on the screen will cause LeScript to print a listing of that directory (4.17).

4.22. END OF TEXT CLEAR-E

The text is scrolled up until the last line appears at the bottom of the screen. The cursor is then moved to the first blank line after the last line of text.

4.23. FILE TEXT TO DISK CLEAR-F

The text which is in the computer's memory and being displayed on the screen will be filed onto the disk according to the file name as it appears in the NAME FIELD of the status line (3.1). To file the text to disk as an "ASCII" file (non-LeScript-standard), the three characters "ASC" must be typed in the WIDTH FIELD prior to hitting CLEAR-F. See Appendix A for details on the differences of these two file types. Once filed, the text will appear back on the screen so you can resume working on it.

4.24. GET TEXT FROM DISK CLEAR-G

There are two ways to invoke this function. You may call a DIRECTORY to the screen (4.21), move the cursor to the desired file name, and hit CLEAR-G. If there is text already in the computer's memory, then the new text will be inserted at the place where your cursor was just before you enacted the DIRECTORY function. If you get a second or third file at this time, they will be inserted following the ones before it. If there was no text in the computer's memory at the time, then the name and width of the first file will automatically be placed in their respective fields in the STATUS LINE (3.1, 3.2). Then hit CLEAR-Z to display the text file on your screen.

The second way to invoke this function is to bypass the DIRECTORY and just type the name of the file on a blank line (see 4.32 to open a blank line) and hit CLEAR-G. The file name will be removed from the text where it was typed, and the file with that name will be read into the computer's memory and inserted into the text at the place where the cursor was.

Note: File names must comply with the rules set forth in your DOS operator's manual, must be left justified on the line in which they were typed, must be no longer than 14 characters, including the extension and drive specification (if used), and must not contain passwords.

4.25. HOME CURSOR CLEAR-H

The cursor is moved to the first column of the first line of the display screen.

4.26. INSERT CLEAR-I

The cursor will blink at a much slower rate to let you know that the INSERT function is active. As you type, all characters to the right of the cursor will be pushed right as characters are inserted at the cursor. As words reach the end of the cursored line they are automatically removed and put on the following line. The INSERT function can be terminated by hitting CLEAR-I a second time.

4.27. REJUSTIFY CLEAR-J

As you type you will notice that LeScript does a marvelous job of keeping all the text lines justified except for the one that the cursor is on. This was done intentionally so that the words on the line of text where you are typing would stay still and not be jumping around every time you hit a key.

This function, however, can be used to totally REJUSTIFY the section of text that is on the screen if you wish to see it in its fully justified form before moving on to another section of text. After hitting CLEAR-J, the text is removed from the screen and returned fully WORD-WRAPPED and LINE-JUSTIFIED.

4.28. KILL FILE ON DISK CLEAR-K

Although this command (CLEAR-K) is invoked in the same way as the TAB function (4.29), it's doubtful that you will get the two confused. The TAB function only works when there is text on the screen, and the KILL FILE function only works when a DIRECTORY is on the screen.

To implement the KILL FILE function, just call the DIRECTORY to the screen (4.21), position the cursor over the file you wish to eliminate from the disk, and hit CLEAR-K. The first time CLEAR-K is struck the "ONCE MORE TO KILL" message will flash on and off in the third STATUS LINE. If you change your mind at this point, just hit any other key to abort. Hitting CLEAR-K a second time will remove the file from the disk. The DIRECTORY is then recalled, showing you that the file is no longer listed in the directory.

This function is especially useful if you need to file away to disk the text you are working on, but the diskette is already full. You can KILL something on the disk to make room for the new text file without ever losing the text in the computer's memory.

4.29. TAB

CLEAR-K

The cursor is moved to the next TAB STOP position, and TAB spaces will be written from the point where the cursor was up to the next TAB STOP. See section 5.4 for the setting of TAB STOPS. The TAB function is what should be used to indent paragraphs.

On proportional-space printers, LeScript precisely calculates TAB distances for each TAB in each line to insure that all the TAB STOPS line up straight down the page.

When a TAB is on the display screen, it will appear as a group of spaces. However, each set of TAB space groups (from one TAB STOP to the next) is changed into a single TAB ASCII CODE (09H) when the file is written to disk. Likewise, when reading a text file from disk, all TAB ASCII CODES are automatically replaced by the appropriate number of TAB characters.

When using the TAB function, it is recommended that you do so only on lines that are set to JUSTIFY LEFT or JUSTIFY BOTH (2.2, 5.11). The meaning of a TAB in a line set to JUSTIFY RIGHT or JUSTIFY CENTER is obscure and the results would be unpredictable.

Note: If this function is used while the DIRECTORY is on the screen, it will be interpreted as the KILL FILE command (4.29) instead of the TAB command.

4.30. DELETE LINE

CLEAR-L

All the text from the cursor to the end of the line will be deleted. If the cursor is in the first column, then the whole line will be deleted, and the lines under the cursor will scroll up to fill the gap.

4.31. MOVE BLOCK

CLEAR-M

Once a block of text is defined (4.19), that block can be moved to any place in your text by simply moving your cursor to that place and hitting CLEAR-M. When the block is moved, the block markers are moved with it. The command can be immediately invoked again, or the block markers can be removed by hitting CLEAR-Q (4.36).

4.32. STORE FILE TO DISK CLEAR-N

Once a block has been defined (4.19), that block can be written onto disk using this function. On a blank line (see 4.33 to open a blank line), write the desired name you wish that block to be filed under, then hit CLEAR-N. A copy of that block will be written to disk under that name. The file name is automatically removed from the text. If you wish to remove the block markers at this time, you may do so by hitting CLEAR-Q (4.36).

Note: File names must comply with the rules set forth in your DOS operator's manual, must be left justified on the line in which they were typed, must be no longer than 14 characters including the extension and drive specification (if used), and must not contain passwords.

4.33. OPEN LINE CLEAR-O

All text lines below the cursor are scrolled down to open one blank line. The cursor is then moved to the first column of the blank line.

4.34. PRINT TEXT CLEAR-P

The text file will be sent to the printer in its final letter-quality form. See section 5 for details on printer initializing and print formatting. Once the text file that is in the computer's memory has been completely PRINTED, the function is terminated and the top portion of the text appears back on the screen. The only exception to this is when there is one or more files in the PRINT CHAIN (4.35). In this case, after the first file is PRINTED, the next file in the PRINT CHAIN is read from disk into memory and treated as if it were part of the previous text file. Or in other words, there will be no gap in the printed page between where one file stopped and the next file in the PRINT CHAIN started. If you wish there to be a gap, we recommend the use of the TOP-OF-FORM command at the end of each file (5.18). The function will automatically terminate when the last file in the PRINT QUEUE is PRINTED, or manually by hitting CLEAR-Z.

4.35. QUEUE FILES FOR CHAINED PRINTING CLEAR-Q

Up to 21 files (13 if screen is 64x16) can be registered in the PRINT CHAIN list by calling the DIRECTORY to the screen (4.21), positioning the cursor over the desired file name, and hitting CLEAR-Q. The names of the files in the PRINT CHAIN will then be entered in First-In-First-Out order as seen on the right side of the screen. Entries may be removed from the PRINT CHAIN by positioning the cursor over the file name in the PRINT CHAIN and hitting CLEAR-L.

Those printer commands which will not carry over from file-to-file and will have to be contained in each queued file are: PRINTER CONTROL CODES (5.20), PROGRAMMABLE FUNCTION CHRACTERS (5.21), and HEADER and FOOTER FORMAT definitions (5.23, 5.24). Page numbering and all other printer commands once set in one file in the PRINTER CHAIN queue are valid through to the last file in that queue.

4.36. REMOVE BLOCK MARKERS CLEAR-Q

This command will remove all the block marker characters (■, 4.19) from the text file that is in memory. Note: If this function is used while the directory is on the display screen, it will be interpreted as the QUEUE FILE FOR CHAINED PRINTING function (4.35).

4.37. SEARCH CLEAR-S

The text will be SEARCHED from the positon after the cursor down to find a string of characters that matches the string of characters you have typed in the SEARCH FIELD (3.6). If found, the cursor will stop at the first character of that string. If not found, the cursor will stop on the next blank line after the last line of your text.

4.38. REPLACE CLEAR-R

If the cursor is positioned at the start of a string of characters that matches the string of characters you have typed in the SEARCH FIELD, those characters, upon hitting CLEAR-R, will be REPLACED with a copy of the string of characters which you have typed in the REPLACE FIELD. If the characters in the SEARCH FIELD do not match those starting at the cursor position, nothing will happen. See 3.7 for more information on how the REPLACE function can also be used for AUTOMATIC TEXT INSERTION.

4.39. TOP OF TEXT CLEAR-T

The portion of your text which is currently on the screen is scrolled off the screen, and the top portion of the text is scrolled onto the screen.

4.40. UNDEFINED CLEAR-U

This function is reserved for future releases of LeScript.

4.41. VIEW TEXT CLEAR-V

Implementing this function will allow the user to VIEW his text in final format with proper indents, page breaks, header lines, footer lines, page number lines, and page numbers installed, non-printing characters (END-OF-SENTENCE characters, BLANKS, and RETURNS) replaced by SPACES, printer command lines removed, and all text fully justified--all without ever printing it on paper. Because of the limitations of the TRS-80 hardware, however, certain things like depth of margin, bold, underline, superscript and subscript cannot be displayed on the screen during VIEWING and must be printed to be seen.

Upon first hitting CLEAR-V, the screen will go blank. Hitting the DOWN ARROW key (↓) will cause the text to scroll forward. Holding this key down will cause continuous scrolling. If desired, you can scroll to a certain place in your text, then hit CLEAR-P, and all the rest of your text, from the last line displayed on your screen through to the end, will be printed on your printer. Terminating the VIEW function is done by hitting CLEAR-Z (4.45).

Note: It is not possible to scroll backwards through the text while implementing the VIEW function.

4.42. DELETE WORD CLEAR-W

All characters from the nearest space to the left of the cursor through the nearest space to the right of the cursor are deleted. All characters on the cursored line to the right of those deleted are then shifted left to fill in the gap. If the cursor is on a SPACE while trying to implement the DELETE WORD function, the word to the right of the space will be deleted.

4.43. EXPANDED CHARACTERS CLEAR-X

When the EXPANDED CHARACTER function is active, all characters typed will appear on the screen as a thin disjointed line (⁞) followed by the character you type. All text entered this way will get printed on your printer as an expanded character, provided your printer has that capability. If your printer is not able to print EXPANDED characters, the character will be printed as a space followed by the normal-width character. Terminating the EXPANDED CHARACTER function is done by hitting CLEAR-X a second time.

Note: The following characters will be displayed on the screen as single-width characters even if the EXPANDED CHARACTER function is active: BLANK characters (▒), END-OF-SENTENCE characters (▬), PRINTER COMMAND characters (▮), RETURN characters (▪), and SPACES ( ).

4.44. SPLIT TEXT CLEAR-Y

If you have a need to insert type at a certain place in your text but you want to do it with a blank screen, then this is the function you want to use. Move the cursor to the place you want to insert the text and hit CLEAR-Y. All text before the cursor scrolls up and off the screen, and all text at and following the cursor scrolls down and off the screen.

Even though this is not a function that you have to terminate, you will want to do something to get your disjointed segments of text into one piece again. This can be done by either scrolling down an excess of one screen then scrolling up the same distance or by hitting CLEAR-J (4.27).

4.45. TERMINATE FUNCTION CLEAR-Z

The following functions are terminated by hitting CLEAR-Z:

4.17	PRINT SCREEN	(CLEAR-@)
4.18	AUTOMATIC SEARCH AND REPLACE	(CLEAR-A)
4.21	DIRECTORY	(CLEAR-D)
4.34	PRINT TEXT	(CLEAR-P)
4.37	SEARCH	(CLEAR-S)
4.41	VIEW TEXT	(CLEAR-V)

4.46. DELETE BLOCK — CLEAR-*

The first time this key is struck the "ONCE MORE TO DELETE" message will flash on and off on the third STATUS LINE. If you change your mind at this point, just hit any other key to abort. Hitting CLEAR-* a second time will cause all the text from the cursor to the next BLOCK MARKER character (4.19) to be deleted.

4.47. RETURN CHARACTER — ENTER

The RETURN character (■) is entered at the cursor, and the cursor is moved to the first column of the next line. Any text which was to the right of the cursor before the ENTER key was struck will also move down to the next line.

Use the RETURN character to terminate a paragraph or a BASIC or an EDTASM statement line or at any time when you want the text on the line below the cursor to stay separate from the cursored line. If the RETURN character is not used to terminate short lines, the LINE JUSTIFICATION function in LeScript will, during scrolling, fill each line with as much text that will fit before starting a new line. RETURN characters can also be used to create blank lines.

Note: The RETURN character is one of the characters which is not displayed while implementing the VIEW TEXT (4.41) or the PRINT TEXT (4.34) functions.

4.48. BLANK CHARACTER — CLEAR-SPACE

A BLANK character (▩) is entered at the cursor. This character is special in that it is treated as a hard character (non-space) by LeScript's LINE JUSTIFICATION function but as a SPACE by it's PRINT TEXT function. It is quite useful in putting extra spaces between words that the LINE JUSTIFICATION function will not close up and yet is displayed as a SPACE while VIEWING (4.41) or PRINTING (4.34) text.

Printing in mono-space mode, the width of the BLANK is the same as all the other characters. However, in proportional-space mode, the width of the BLANK will always be 1/CHARACTER DENSITY. For example, if you specify a CHARACTER DENSITY of 12 characters per inch (5.14), a BLANK will be printed as 1/12 of an inch wide. This width, once established by the CHARACTER DENSITY specification, will never decrease or increase the way a SPACE would in justifying a line to both margins.

4.49. END-OF-SENTENCE CHARACTER CLEAR-ENTER

The END-OF-SENTENCE character (■) is entered at the cursor. Use this character to create an additional amount of space between sentences. If an END-OF-SENTENCE character is used after a sentence, a space should be typed after that character. This character functions exactly like the BLANK character (▩) except for one respect. If this character falls at the end of a line during PRINTING (4.34) or VIEWING (4.41) and the line justification is JUSTIFY BOTH (2.2, 5.11), this character will get temporarily removed and the line rejustified to extend out to meet the right margin.

4.50. STATUS LINE HOME CLEAR-=

The cursor is moved to the NAME FIELD of the top STATUS LINE.

4.51. CLEAR CLEAR-0

The first time this key is struck the "ONCE MORE TO CLEAR" message flashes on and off on the third STATUS LINE. If you change your mind at this point, just hit any other key to abort. Hitting CLEAR-0 a second time will cause the screen and the memory to clear and the STATUS LINES to re-initialize.

4.52. EXIT TO DOS READY SHIFT-CLEAR-0

The first time this key is struck the "ONCE MORE TO EXIT TO DOS" message flashes on and off on the third STATUS LINE. If you change your mind at this point, just hit any other key to abort. Hitting SHIFT-CLEAR-0 a second time will cause LeScript to exit and give control of the computer back over to DOS. After exiting, it is possible to re-enter LeScript with your text still intact (provided no DOS routine has over-written it) by typing: LESCRIPT *<ENTER>. You usually re-enter LeScript this way after an accidental system reset or power glitch, provided the text is still intact in your computer's memory.

4.53. SUPERSCRIPT CLEAR-1

To make a character(s) print out in SUPERSCRIPT (1/2 line feed above the base line), just move the cursor to that feed above the base line), just move the cursor to that character(s) and hit CLEAR-1. The character will then alternately blink between the character itself and a small graphics block that is 1/2 line feed above the base line (▀). To reverse this function, move the cursor over the SUPERSCRIPTED character(s) and hit CLEAR-1 again. The character will then stop blinking.

Note: Not all printers have the ability to print SUPERSCRIPTS.

4.54. SUBSCRIPT CLEAR-2

To make a character(s) print out in SUBSCRIPT (1/2 line feed below the base line), just move the cursor to that character(s) and hit CLEAR-2. The character will then alternately blink between the character itself and a small graphics block that is on base line (▄). To reverse this function, move the cursor over the SUBSCRIPTED character(s) and hit CLEAR-2 again. The character will then stop blinking.

Note: Not all printers have the ability to print SUBSCRIPTS.

4.55. BOLD CLEAR-3

To make a character(s) print out in BOLD, just move the cursor to that character(s) and hit CLEAR-3. The character will then alternately blink between the character itself and the asterisk character (*). To reverse this function, move the cursor over the BOLD character(s) and hit CLEAR-3 again. The character will then stop blinking.

Bolding can be combined with other character enhancement functions on the same character (SUPERSCRIPTS, SUBSCRIPTS, UNDERLINE, ITALICS, and CHARACTER PAUSE). All functions that are active on a character will blink in turn their own function's enhencement symbol.

Note: Not all printers have the ability to print BOLD.

4.56. UNDERLINE CLEAR-4

To make a character(s) print out with an UNDERLINE, just move the cursor to that character(s) and hit CLEAR-4. The character will then alternately blink between the character itself and the underline bar character (_). To reverse this function, move the cursor over the UNDERLINED character(s) and hit CLEAR-4 again. The character will then stop blinking.

Note: Not all printers have the ability the print UNDERLINES.

4.57. ITALIC CLEAR-5

To make a character(s) print out in ITALICS, just move the cursor to that character(s) and hit CLEAR-5. The character will then alternately blink between the character itself and the slash character (/). To reverse this function, move the cursor over the ITALIC character(s) and hit CLEAR-5 again. The character will then stop blinking.

Note: Not all printers have the ability the print in ITALICS.

4.58. CHARACTER PAUSE CLEAR-6

To make your printer pause at any given place in your text, like to change a print wheel or the color of your ribbon, just move the cursor to the character(s) you wish the printer to stop before and hit CLEAR-6. The character will then alternately blink between the character itself and the dash character (-). To reverse this function, move the cursor over the character(s) and hit CLEAR-6 again. The character will then stop blinking.

4.59. UNDEFINED CLEAR-7

This function is reserved for future releases of LeScript.

4.60/ RENUMBER CLEAR-8

This function only works when the text being edited is a BASIC or EDTASM file. The line numbering starts at 10 and counts up by 10's. In both the BASIC and EDTASM file, the renumbering is done by stripping off all leading numeric characters on each line, then reinserting the appropriate number value. However, EDTASM files will always end up with five-digit line numbers, but the line numbers on the BASIC files will only be as many digits as they have to be. For example: Line number 10 will be "00010" in EDTASM file and "10" in BASIC files.

Note: This function will only reassign line numbers. It will not change the GOTO and GOSUB statements in your BASIC text files.

4.61. KEYBOARD CLICK CLEAR-9

The use of the cassette relay within the computer to produce a gentle click when a key is struck is to give the operator an audible feedback mechanism reassuring him that the key-stroke has been seen and processed. Hitting CLEAR-9 will activate this function. Hitting CLEAR-9 again will deactivate this function.

4.62. EXTENDED CHARACTER SET KEYS

There are many characters that your computer can generate (and your printer might be able to print) that are not found on the keys of your keyboard. LeScript lets you access several of these characters by holding down the SHIFT key and the CLEAR key and hitting one of the other keys. A complete list of these characters is on the following page.

All of these characters can also be "redefined" or programmed to output any sequence of characters and printer control codes to the printer at the time your text is printed. For example: say your printer cannot print "ñ", but it can print "n", then back space, then print "~". OK, then with LeScript you can program the "ñ" character that appears on your display screen to output "n", backspace, "~". There are a total of 54 such characters that can be programmed in this manner (these 38 and 16 others). See section 5.21 for the list of the other characters that can be redefined and for more information on how this is done.

SHIFT-CLEAR-<key>	character
(	[(↑ Model I)
)	] (← Model I)
<	{
>	}
− (dash)	_ (underline)
@	` (Æ Model I)
/	\ (↓ model I)
*	~

Model III, Model 4, and MAX-80 only:

A	£
B	|
C	é
D	Ü
E	Å
F	¬
G	ö
H	Ø
I	ù
J	ñ
K	`
L	ā
M	R̄
N	Ä
O	Ã
P	Ñ
Q	ȯ
R	ø
S	Ö
T	ß
U	ü
V	õ
W	‡
X	ä
Y	à
Z	ȧ
1	§
2	É
3	Æ
4	Ç
5	~

5. PRINTER COMMANDS

PRINTER COMMANDS tell LeScript how to format the text that is to be printed. Margin size, right and left side text indenting, line spacing, position of header line, and character pitch are examples of the many PRINTER COMMANDS available to you.

PRINTER COMMANDS are found in what is known as PRINTER COMMAND LINES. Each PRINTER COMMAND LINE must begin with a PRINTER COMMAND CHARACTER (▮)(4.16) and end with a RETURN character (▪)(4.47), and each PRINTER COMMAND within the line must be separated from the other PRINTER COMMANDS with commas (,). These PRINTER COMMANDS are visible in the text while the text is being edited but are not displayed while the text is being PRINTED (4.34) or VIEWED (4.41).

In the following pages, all the various PRINTER COMMANDS are summarized and examples of each are given. An expression showing the standard format of each PRINTER COMMAND is listed to the right of each PRINTER COMMAND name. The upper case characters in the expressions are the "constants", the characters which define the PRINTER COMMAND type, and will not change. The lower case characters in the expressions are the "variables", where you, the user, can specify a value or a string of characters that will be associated with that command from that point on or until the command is used again somewhere else in the text. Even though in the standard command format expressions the constants are upper case and the variables are lower case for clarity in explanation, the constants and the variables of any PRINTER COMMAND can actually be typed in either upper or lower case--whichever is easiest for you.

Even though there are 30 different PRINTER COMMMANDS in all, you need not let that number overwhelm you. LeScript is indeed a very simple word processing system to use. You can actually create and print out text <u>without ever using any PRINTER COMMANDS</u>. Try it to prove it to yourself. The internal defaults for each function will take over when the PRINTER COMMANDS are absent. But if there is something special you need a word processor to do, it is altogether likely that LeScript can do it using one of the following PRINTER COMMANDS.

Caution: There is no error checking on PRINTER COMMANDS. Be careful to follow the examples--don't leave anything out, don't add anything, and don't change anything, unless it's stated that you can. Improper PRINTER COMMANDS will result in a misinterpretation of that command and often the one following it, as well.

Note: The periods in the examples (...) are to show which commands can be followed by or preceded by other commands, and which ones cannot. They are not to be used in actual commands.

5.1. PRINTER TYPE Kn

Use this printer command to specify which type of printer you own. This is the only way LeScript knows which control codes to use in order to control the various functions of your printer. The printers which are supported by LeScript are listed below along with the printer type commands used to specify them. The default for this function is "K0".

K0	OLIVETTI Bytewriter, and any other such printer that has no special print functions, or any printer which is not mentioned in the list below. Underline and Bold printing can be achieved on any printer using "K0" as long as the automatic linefeed of the printer can be disabled and the "LF" printer command (see 5.2) is used.
K1	Radio Shack Daisy Wheel II, DWP-210, and DWP-410 (mono-space operation).
K2	Radio Shack Daisy Wheel II (proportional-space operation)
K3	Radio Shack DWP-210 and DWP-410 (proportional-space operation).
K4	C.Itoh Starwriter FP-1500, C.Itoh Starwriter F-10, Brother HR-15, Diablo 630, Diablo 1610, JUKI 6100, Dataproducts DP-55 RO, Silver-Reed EXP-500, EXP-550, BMC-501, NEC 3515, NEC 3525, NEC 5525, NEC 7715, NEC 7725, Qume Sprint and most other daisy wheel printers (mono-space operation).
K5	C.Itoh Starwriter FP-1500, C.Itoh Starwriter F-10, Brother HR-15, Diablo 630, Diablo 1610, JUKI 6100, Dataproducts DP-55 RO, Silver-Reed EXP-500, EXP-550, BMC-501, NEC 3515, NEC 3525, NEC 5525, NEC 7715, NEC 7725, Qume Sprint and most other daisy wheel printers (proportional-space operation).
K6	NEC 3510, NEC 3520, NEC 5520, NEC 7710, NEC 7720, (mono-space operation).

k7	NEC 3510, NEC 3520, NEC 5520, NEC 7710, NEC 7720, (mono-space operation).
K10	DAISY WRITER 1500 and DAISY WRITER 2000 (mono-space operation).
K11	DAISY WRITER 1500 and DAISY WRITER 2000 (proportional-space operation).
K12	DTC 3802 (mono-space operation).
K13	DTC 3802 (proportional-space operation).
K14	Brother HR-1 and COMREX Comwriter CR-1 (mono-space operation).
K16	SMITH-CORONA TP-1 (mono-space operation).
K30	EPSON MX-80 (mono-space operation).
K32	EPSON MX-80 with GraftraxPlus, EPSON MX-100 with GraftraxPlus, EPSON RX-80, EPSON FX-80, EPSON FX-100, Star Micronics Gemini-10, Star Micronic Gemini-10X and Star Micronics Gemini-15 (mono-space operation).
K33	EPSON MX-80 with GraftraxPlus and EPSON MX-100 with GraftraxPlus, (proportional-space operation).
K35	EPSON RX-80, EPSON FX-80, and EPSON FX-100 (proportional-space operation).
K37	Star Micronics Gemini-10, Star Micronic Gemini-10X and Star Micronics Gemini-15 (proportional-space operation).
K40	OKIDATA Microline 80, Microline 82A, and Microline 83A (mono-space operation).
K42	OKIDATA Microline 84, and Microline 92 (mono-space operation).

K43 OKIDATA Microline 84, and Microline 92 (proportional-space operation).

K50 C. ITOH Prowriter 8510, C. ITOH Prowriter 8515, NEC PC-8023A-C, PMC-8510 TEC 8500R, and TEC DMP-85 (mono-space operation).

K51 C. ITOH Prowriter 8510, C. ITOH Prowriter 8515, and PMC-8510 (proportional-space operation).

K53 NEC PC-8023A-C (proportional-space operation).

K55 TEC 8500R and TEC DMP-85 (proportional-space operation).

K60 Radio Shack Line Printer IV, Centronics 737, and Centronics 739 (mono-space operation).

K61 Radio Shack Line Printer IV, Centronics 737, and Centronics 739 (proportional-space operation).

K62 Radio Shack Line Printer VI (mono-space operation).

K64 Radio Shack Line Printer VIII (mono-space operation).

K65 Radio Shack Line Printer VIII (proportional-space operation).

K66 Radio Shack DMP-100 (mono-space operation).

K68 Radio Shack DMP-200, DMP-400, and DMP-500 (mono-space operation).

K69 Radio Shack DMP-200, DMP-400, and DMP-500 (proportional-space operation).

K70 Radio Shack DMP-2100 (mono-space operation).

K71 Radio Shack DMP-2100 (proportional-space operation).

K80	Toshiba P1350 (courier print) (mono-space operation).
K81	Toshiba P1350 (courier print) (proportional-space operation).
K82	Toshiba P1350 (elite print) (mono-space operation).
K83	Toshiba P1350 (elite print) (proportional-space operation).
K86	IDS Microprism 480 (mono-space operation).
K87	IDS Microprism 480 (proportional-space operation).
K90	QANTEX 7030 (mono-space operation).
K92	INFOSCRIBE 1000 (mono-space operation).
K94	BASE-2 (mono-space operation).
K96	MPI PrintMate 88, 99, and 150 (mono-space operation).

Example: ("..." means possible other commands)

```
■...,K53,...■
```

This command specifies that the text is going to be printed on an NEC PC-8023A-C printer in proportional-space mode.

Note: it is recommened that this command be the first printer command used in your text file, preceded only by the BAUD RATE printer command (if it is used, see 5.3).

5.2. AUTOMATIC LINE-FEED INSERTION

With LeScript, you have the option of choosing between having each printed line terminated by a CARRIAGE-RETURN and a LINE-FEED or just a CARRIAGE-RETURN. Which you choose will depend on whether or not your printer can be configured to accept bth the CARRIAGE-RETURN and the LINE-FEED for the line termination and by the printer configuration requirements of your other programs. The default for this function is CARRIAGE-RETURN only.

Note: If your printer is one that is incapable of reverse LINE-FEEDS, we recommend that this command be used and the printer be configured to accept both the CARRIAGE-RETURN and the LINE-FEED from the host computer. Else, functions that normally require overstriking, such as SUPPRESS LINE-FEED, BOLD, and UNDERLINE, may not work properly on your printer.

Example: ("..." means possible other commands)

```
    ■...,LF,...■
```

In this exaple, all the text lines will be terminated by sending a CARRIAGE-RETURN and a LINE-FEED to the printer.

5.3. BAUD RATE FOR SERIAL PRINTERS

If this command is specified, LeScript assumes that your printer is connected to the serial port of your computer. If this command is not specified, LeScript assumes that your printer is connected to the parallel printer port on your computer. The 8 different forms of this command and the baud rates which they represent are listed in the table below.

R1	110 baud
R2	150 baud
R3	300 baud
R4	600 baud
R5	1200 baud
R6	2400 baud
R7	4800 baud
R8	9600 baud

Note: The baud rate inside your printer has to be set the same. Consult the operator's manual that came with your printer. For smoothest and fastest printing, we recommend that you set your printer at the highest BAUD rate that it is capable of running at.

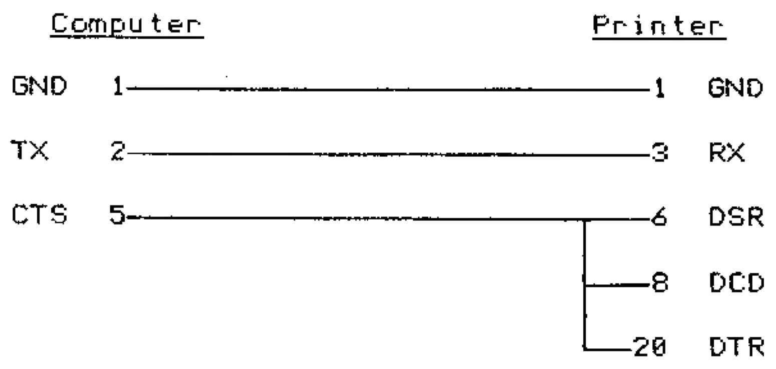
Caution: It is imperative that, if this command is used, it is to be the first printer command in the text.

Example: ("..." means possible other commands)

■...,R4,...■

This command specifies a serial printer running at 600 baud, rather than a parallel printer.

For correct operation, the RS-232 cable between computer and printer must be wired as follows:



5.4. SETTING OF TABS

TABa;b;c;d;e;f.....

This printer command is used to set the tab stops for both editing and printing. The letters in the expression (a-f...) are the tab stop locations relative to the left most column of the line. These must be specified in ascending order within the command. As many as 15 tab stops can be specified. This command will become active on the screen for editing once entered and CLEAR-T (4.39) has been struck. The default for this command is:

```
▮TAB6;14;22;30;38;46;54;62;70;78;86;94;102;110;118▪
```

Note: All the stops within the command are separated by semicolons ";" **not** commas.

Caution: For proper operation, no more than one TAB STOP command should be made per file, and it should be located within the text file ahead of any printed text.

Example: ("..." means possible other commands)

```
▮...,TAB10;17;21;28;35;42;49;56,...▪
```

The first tab stop is 10 columns in from the left side, the second is 17 columns in from the left side, ..., the last one is 56 columns in from the left side.

5.5. INDIVIDUAL SHEET PAUSING

I

If you are feeding your printer with individual sheets instead of a continuous roll or a box of fan fold paper, then you will want to use this command. Each time the print head reaches the line number on the page that you have specified as the length of the paper (5.6), the printing will pause. This gives you the opportunity to remove that sheet from the printer and put a new one in without losing any text. To restart the printer at the point where you left off, either hit the space bar on the keyboard or turn the on-line switch off and back on. This action will have to be taken before LeScript will continue printing any sheet of that file--even the first sheet. The default for this function is that the individual sheet pausing is not active.

Example: ("..." means possible other commands)

```
▮...,I,...▪
```

This command activates the single sheet pausing function.

5.6. SHEET SIZE SET

Yn

This command is used to set the SHEET SIZE of the paper on your printer. The "n" in the expression is the length in number of lines. The default for this function is "Y66".

Example: ("..." means possible other commands)

```
■...,Y48,...■
```

This specifies a SHEET SIZE of 48 lines long, or 8 inches assuming 6 lines per inch spacing (standard).

5.7. TOP TEXT LINE SET

Tn

This printer command establishes on which line, counting from the top of the page, the body of the text will begin. The "n" in the expression is that line number, 1 being the very top of the page. The default for this function is "T7".

Caution: The number that you assign for the TOP TEXT LINE must be less than or equal to the number you assign for the END TEXT LINE (5.8), and must be greater than 0.

Example: ("..." means possible other commands)

```
■...,T1,...■
```

In this example the text body will begin printing on the top line of the page.

5.8. END TEXT LINE SET En

This command is just like the TOP TEXT LINE SET printer command (5.7), except that it specifies on which line the last line of the text body will be printed. The default for this function is "E62".

Caution: The number that you assign for the END TEXT LINE must be greater than or equal to the number you assign for the TOP TEXT LINE (5.7), and must be less than or equal to the number you assign for the SHEET SIZE (5.6).

Example: ("..." means possible other commands)

```
    ■...,E57,...■
```

In this example the text body will end approximately 9.5 inches below the top of the page, assuming 6 lines per inch spacing (standard).

Example: ("..." means possible other commands)

```
    ■...,T1,E1,Y1,...■
```

This set of commands will allow you to print out any number of text lines without any paper waste at the top or bottom (no top or bottom margin.

5.9. LEFT MARGIN MOn,MEn

This printer command will consist of either "MO" followed by a value "n" to specify the width of the left side margin on odd-numbered pages, or "ME" followed by a value "n" to specify the width of the left margin on the even-numbered pages. Both must be specified or the default (MO8,ME8) will be assumed. The unit of measure for "n" is either tenths of an inch (if you are printing in proportional-space mode), or character widths (if you are printing in mono-space mode).

Example: ("..." means possible other commands)

```
■...,MO12,...,ME6,...■
```

In this example, the left margin width is 1.2 inches on the odd-numbered pages and .6 inches on the even-numbered pages, if you are printing in proportional-space mode. Otherwise, the left margins in this example will be 12 character widths and 6 character widths, respectively. The actual width of the character depends on the type of printer you own and which character pitch you have set the printer for (5.13).

5.10. LINE SPACING Ln

The "n" in this expression is to be set to the desired spacing of the text lines--"1" for single spaced-lines, "2" for double-spaced, etc. No value for "n" less than 1, greater than 10, or greater than the number you specified as the SHEET SIZE (5.6) is allowed. The default for this function is "L1".

Note: Improper matching of the AUTOMATIC LINE-FEED INSERTION printer command (5.2) with the configuration of your printer will result in improper line spacing. If you are getting double spacing when you specified single spacing, it is probably because you are using the "LF" command (5.2) and shouldn't be. If all the text is getting printed on the same line and there are no line-feeds at all, it is probably because you are not using the "LF" command and should be.

Example: ("..." means possible other commands)

```
        █...,L2,...▪
```

In this example the text lines will be printed double-spaced.

Note: to get 1.5 line spacing use:

```
        █...,L1,CLxx,Y44,...▪
```

The "xx" are printer control codes (in hexidecimal) which will cause your printer to advance a 1/2 forward line-feed. Consult your printer manual. And the "Y44" replaces the "Y66" for standard 11 inch paper.

5.11. JUSTIFY MODE SET

This printer command lets you specify which of four possible LINE JUSTIFICATION modes you can have the text to be printed in.

JL	Justify text lines to the left margin.
JR	Justify text lines to the right margin.
JB	Justify text lines to both the left and right margins.
JC	Center the text between the left and right margins.

These four different commands will not only dictate the way the text will justify on the page as it is being PRINTED (4.34), but will also cause the text to justify accordingly on the screen while the text is being VIEWED (4.41), as well as while you are editing it. The line containing the command must, however, be scrolled off the screen then brought back onto the screen by scrolling <u>up</u> in order to set the LINE JUSTIFICATION mode for that portion of text while you are editing it. Hitting CLEAR-T (4.39) is one way of achieving these results. The default JUSTIFY mode for PRINTING, VIEWING, and editing is "JL".

Note: While PRINTING or VIEWING text that is set to JUSTIFY RIGHT (JR), lines terminated by RETURN characters (■) will be justified fully to the right margin after the RETURN character has been stripped.

Example: ("..." means possible other commands)

```
▌...,JL,...■
Mares eat oats,■
▌...,JR,...■
And does eat oats,■
▌...,JB,...■
And little lambs
▌...,JC,...■
eat ivy.■
```

This will be printed:

```
Mares eat oats,
     And does eat oats,
And     little     lambs
         eat ivy.
```

5.12. TEXT INDENTING

These two printer commands can be used to achieve text widths which are less than the maximum (set in the WIDTH FIELD, see 3.2), by specifying an additional amount of indenting on the left or right sides or both. The "n" in the expressions is the number of character widths of indenting desired. This is often useful in highlighting certain paragraphs by causing them to be narrower than the rest of the text body. Full justification is maintained on all indented portions of text. The default for this function is "IL0,IR0".

Note: The LEFT SIDE TEXT INDENT is in addition to the TEXT LEFT MARGIN as described in section 5.9.

Example: ("..." means possible other commands)

```
▒▒▒The  following  letter was received by our
branch office in Chicago.■
■...,IL5,IR5,GR1,...■
     Dear Sirs,■
     ▒▒▒Thank  you  for  your  help  in
     locating the parts that I needed. I
     will  be  doing  more business with
     you in the future.■
                    Sincerely,■
                    Jack Thomson■
■...,IL0,IR0,GR1,...■
▒▒▒Congratulations,  Chicago!   It's  happy
customers  like  this  one  that have kept us
going strong.■
```

This will be printed:

```
    The  following  letter  was  received  by  our
branch  office  in  Chicago.

       Dear  Sirs,
          Thank  you  for  your  help  in
       locating  the  parts  that  I  needed.  I
       will  be  doing  more  business  with
       you  in  the  future.
                    Sincerely,
                    Jack Thomson

   Congratulations,  Chicago!   It's  happy
customers  like  this  one  that  have  kept  us
going  strong.
```

5.13. CHARACTER PITCH   Qn

The "n" in this expression is a single digit number 1-8. It is used to express the relative width of the character. The higher the number "n", the wider the characters are printed. Since CHARACTER PITCH is the inverse of the width, it stands that the lower values for "n" generate higher CHARACTER PITCH values.

command	CHARACTER PITCH
Q1	17
Q2	15
Q3	12
Q4	10
Q5	8.5
Q6	7.5
Q7	6.0
Q8	5.0

Not all printers can print all these CHARACTER PITCHES, LeScript's printer drivers are programmed to set the printer you are using to the whatever pitch is closest to the pitch you specify in this command. Q5, Q6, Q7, and Q8 should not be used on daisy wheel printers.

5.14. CHARACTER DENSITY

This printer command is only needed if you are printing in proportional-space mode. This command gives you the capability of deciding how tightly packed you want a line of characters to be printed. The "n" in the expression is the characters-per-inch density, and can be any value from 6.0 to 20.0, in increments of .1 characters per inch.

From this density figure you can determine the actual length of the line in inches knowing the length of the line in characters. Say you have specified your text lines to be 64 characters long, max (3.2), and you want that to print out to exactly 5 inches. Using a little simple math you would determine that this translates to a CHARACTER DENSITY of 12.8 characters per inch. In LeScript, the default CHARACTER DENSITY for most of the dot matrix printers it supports is 13.0 characters per inches. On the daisy wheel printers its 10.0 characters per inch. These are, by the way, just about as tightly as you normally can pack a line on each of the given printers and still have some noticeable amount of space between the words. This manual was printed with a CHARACTER DENSITY of 11.0 characters per inch.

Note: If you are using printer command Q5, Q6, Q7, or Q8 (5.13) the characters will be printed at half the density you specify.

Caution: If character densities are specified which exceed 10.5 on daisy wheel printers or 14.0 on dot matrix printers, you may see one or more lines in which the characters did not justify properly. If this should ever happen it will be necessassary to specify a lower value for the character density.

Example: ("..." means possible other commands)

```
    █...,D13.8,...■
    Jack and Jill ran up a hill.■
    █...,D6.5,...■
    Jack and Jill ran up a hill.■
```

This will be printed:

```
    Jack and Jill ran up a hill.
    J a c k   a n d   J i l l   r a n   u p   a   h i l l .
```

Note: Each "Kn" command (5.1) will cause LeScript to revert back to the default density for that printer, unless a new Density command follows each "Kn" command.

5.15. SUPPRESS LINE-FEED XL

Use this printer command if you ever want to over-print one line with another line. This command only affects the text line that immediately follows it and none other. This command will only work on printers that have the AUTO-LINE-FEED disabled, and/or on printers which are able to perform reverse line-feeds.

Example: ("..." means possible other commands)

```
■...,XL,...■
The current is not to exceed 6.300 amps.■
▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒//■
```

This will be printed:

```
The current is not to exceed 6.3ØØ amps.
```

5.16. TURN PRINTING OFF AND ON

Use these two printer commands in cases when you have a very large text file, but you want only certain parts of it to be printed. These commands can be used to delimit comment statements throughout your text, as well.

All printer commands that fall between the "OFF" and "ON" commands, except for PRINTER CONTROL CODES (5.20), will be processed, even though the text is treated as if it wasn't there.

Note: Using the "ON" and "OFF" printer commands will not cause gaps in text while the text is being PRINTED on the page or VIEWED on the screen. The delimited text is treated like it isn't even there.

Example: ("..." means possible other commands)

```
Since the meeting on Tuesday, orders have
been on a steady increase, employee morale
is better, and the business is doing great.■
▮...,OFF,...■
Maybe I should say more about the meeting.■
▮...,ON,...■
So what I have decided to do is give
every employee a 10% raise in salary.■
```

This would get printed:

```
Since the meeting on Tuesday, orders have
been on a steady increase, employee morale
is better, and the business is doing great.
So what I have decided to do is give
every employee a 10% raise in salary.
```

5.17. GO TO LINE NUMBER Gn

Upon receipt of this command, LeScript will hold off sending any text to the printer. Instead, line feeds will be sent to the printer until the print head reaches the line number on the page specified by the "n" in this command. At this time, the printing of the text resumes. This command is very useful for leaving a large blank area on the paper without putting a lot of RETURN characters in your text. This command can also be used as a TOP-OF-FORM command, as the example shows. If the print head passes any HEADER lines (5.23) or FOOTER lines (5.24) on the way to its "Go To" destination, they will be printed.

Example: ("..." means possible other commands)

```
▮...,G20,...▪
```

This will cause the printer to advance the paper until the print head is at the 20th line of the page.

5.18. TOP-OF-FORM G1

As you have guessed, this is the GO TO LINE NUMBER command (5.17) with a "1" put in for the line number "n". Use it any time you wish nothing more to print on the current page (except for the HEADER and FOOTER lines) and you want printing to start again at the top of the next page.

5.19. GO RELATIVE GRn

This printer command is very similar to the GO TO LINE NUMBER command (5.17) except that it allows you to specify a desired number of blank lines without knowing what line number you are currently on, or which one you're going to.

Example: ("..." means possible other commands)

```
      Peace on Earth,■
      ■...,GR3,...■
      And good will toward men.■
```

This will be printed:

```
      Peace on Earth,



      And good will toward men.
```

Most printers can do more than just print characters. Some can roll the paper backwards, set vertical tabs, change type styles, etc. These are all controlled and communicated to the printer by what is known as PRINTER CONTROL CODES (ASCII values usually less than 20H). If you look in the operator's manual that came with your printer, you will find a list of these special capabilities and the PRINTER CONTROL CODES used to make them work.

If you own one of the printers listed in section 5.1 and have used the proper "Kn" specification, you can have access to most of these special capabilities through the use of the appropriate printer commands. However, if you have a different model printer, or just want to have direct control of your printer's special abilities, LeScript gives you the freedom to do so using this printer command. All of the special type fonts of the Epson printers can be achieved using this function (see second example).

The "aabb..." in either of these two expressions is a set of one or more (not more than a line full) two-digit hexidecimal numbers (with no separation). The "Xn" combination in either of these two expressions is optional and only needed if you wish the string of PRINTER CONTROL CODES to be sent to the printer a repeated number of times. The "n" is any decimal value from 2 to 65535. If an "L" follows the "C", as it does in the second expression, this printer command will be executed at the beginning of <u>every</u> line. If no "L" follow the "C", as in the first expression, this printer command will be executed but <u>once</u>. Each "CL..." printer command in the text will cancel any preceding "CL..." commands.

Example: ("..." means possible other commands)

```
▮...,C0AX20,...▪
```

This command will send 20 line feeds (0AH) to the printer.

```
▮...,CL1B45,...▪
```

On an Epson printer, this command will cause all of the following text to be printed in the "Emphasized" type font.

5.21. PROGRAMMABLE FUNCTION CHARACTERS £(n)'xaa'y'zbb...

The PROGRAMMABLE FUNCTION CHARACTERS command is in many ways similar to the PRINTER CONTROL CODE printer command (5.20). However, it goes one step further in that it allows you to send to the printer a string of ASCII characters and printer control codes <u>mid-line</u>. One programmable FUNCTION <u>character</u> is used in the line of text to "locate" the string (4.17), while a previous PROGRAMMABLE FUNCTION <u>printer command</u> "defines" that character.

THE £ in the expression is any one of 54 different characters. The list includes all characters in the EXTENDED CHARACTER SET (4.62, if your computer can generate them), and any of the following:

!?"#$%&'()<>*+-=/

The "n" in the expression is an optional width value for the character thus defined and is only necessary if printing in proportional-space mode with text justified to the right and left margins. This tells LeScript how much room, in proportional-space pixel widths, this special character is going to require. Any value for "n" from 0 to 254 is allowed. If the "(n)" is used, it must follow right after the PROGRAMMABLE FUNCTION character within the printer command line. If the "(n)" is omitted from the expression, then the normal proportional-space width of the first ASCII character found in the string is assumed. If there are no ASCII characters in the string, a width of 0 is assumed.

The "'x", "'y", and "'z" in the expression are the ASCII characters. Any single character preceded by an apostrophe (') is considered in this command to be an ASCII character. The "aa" and "bb" in the expression are two-digit hexidecimal numbers. These will most often be printer control codes which cannot be expressed as ASCII characters. However, even ASCII characters can be expressed as two-digit hexidecimal numbers if you choose. These can be located in any order and in any amount throughout the expression.

This is the only printer command that can be more than one display line long. See the last example in this section. Follow these rules for the PROGRAMMABLE FUNCTION CHARACTER printer commands that have to be longer than one display line:

1. Each PROGRAMMABLE FUNCTION printer command line should start with the PRINTER COMMAND character (▮), but only the first line should contain the PROGRAMMABLE FUNCTION character.
2. Only the last line of the printer command should be terminated by a RETURN character (▪) or a comma (,).
3. Do not break the line in a place that would split a pair of characters which represents an ASCII character ('x) or a printer control code (aa).

The following are examples of how this function can be used. Keep in mind that each one of these examples was printed using an NEC PC-8023A-C printer. If you have a different type of printer, the concept will be the same; but the implementation will have to conform to your own printer's specifications. Study your printer manual to learn the printer control code representations of graphic characters, extended symbols, foreign language characters, control of type style enhancements, fonts, and the control of print head and carriage movements. Once learned, you could be using the PROGRAMMABLE FUNCTION command to create countless symbols, shapes, borders, pictures--you name it.

1. Access to foreign characters:

```
        █+1B26´J1B24▪
        The circumference of a circle is 2+.▪
printed:
        The circumference of a circle is 2π.
```

2. Access to symbols:

```
        █#1B23´I1B24,?1B23´H1B24▪
        A bid of 3# cannot follow a bid of 3?.▪
printed:
        A bid of 3♠ cannot follow a bid of 3♣.
```

3. Access to print enhancements.

```
        █<1B58´u,>´e1B59▪
        This is another way to <nderlin> a word.▪
printed:
```

This is another way to <u>underline</u> a word.

4. Access to dot graphics (justified in proportional-space).

```
        █$(25)1B´S´0´0´2´5480024001200090012002400480024
        █0012000900120024048▪
        █%(27)1B´S´0´0´2´71C0014001400140014001400140014
        █0022004100410041002200 1C▪
              While on vacation in Africa I discovered a
        small piece of pottery with two strange marks on
        it.  One was a wavy set  of  lines  $,  and  the
        other  was  a circle with a kind of handle %.  I
        have been looking for a year for someone who can
        tell me its meaning.▪
printed:
              While on vacation in Africa I discovered a
        small piece of pottery with two strange marks on
        it.  One was a wavy set of lines ≈, and the
        other was a circle with a kind of handle ⊸.  I
        have been looking for a year for someone who can
        tell me its meaning.
```

5.22. LEGAL COURTROOM NUMBERING N

When this command is used, the lines of your text will be automatically numbered as they would on a courtroom transcript. The first line of each page will be numbered "1" (no quotes), the second line "2", etc. The two digits of the number will be printed in the first and second column of the page. Spacing between the line number and the text is done using the LEFT MARGIN command (5.9).

5.23. BEGIN HEADER FORMAT

Headers, by definition, are any lines of text that fall between the top line of your page (line 1) and the line you have assinged for the top line of the text body (5.7). FOOTERS (see section 5.24), by definition, are any printed lines of text that fall between the last text line (5.8) and the sheet size line (5.6).

LeScript allows you to define different HEADERS and different FOOTERS for both the odd- and even-numbered pages. LeScript also gives you the flexibility to format your HEADERS and FOOTERS virtually any way you could want, with several character pitches, mixed justification modes, and even automatically inserted line numbers anywhere you want them.

You should type your HEADER and FOOTER formats at the beginning of your text file, just before or just after the main formatting commands for your text body. You would format each HEADER and FOOTER (odd-page and even-page) in the same way you would format the text body, using the same printer commands as described on the preceding pages. Once the format for your HEADERS and FOOTERS has been set, they will always print out just that way each time, unless or until you redefine the HEADERS or FOOTERS later on in your text file, which is permissible.

How does LeScript tell the difference between those printer commands which define the format of your text versus those which define the format (and content) of your HEADERS and FOOTERS? Simple, HEADER and FOOTER format data begin with the BEGIN HEADER (or FOOTER) command (HO, HE, FO, FE) on a line by itself, followed by the format data lines (text and printer commands), followed finally by the END HEADER/FOOTER FORMAT printer command (Z) on a line by itself.

Example: ("..." means possible other commands)

```
        ▮HO▪
        ▮JC,MO8,GR4,Q4▪
        TOPICAL REVIEW▪
        Page ##
        ▮Z▪
        ▮HE▪
        ▮JB,ME8,GR4,Q4,L2▪
        Page▒##                                   A▒Space▒In▒Time▪
        ▮JC▪
        By Eli Thornbird▪
        ▮Z▪
```

The Header on page 1 (an odd-numbered page) will print out on line 4 and will look like:

```
                         TOPICAL REVIEW
                             Page 1
```

The HEADER on page 2 (an even-numbered page) will print out starting 4 lines from the end of the text body and will look like:

```
    Page 2                                    A Space In Time

                        By Eli Thornbird
```

You will also notice form the example, that a group of "#" characters is used within the HEADER and FOOTER formats to indicate where you want LeScript to write the page number. Always use enough of the "#" characters in each grouping to match the number of digits in your highest page number. In other words, if your highest page number is going to be 103, then each grouping of "#" characters should be at least 3 characters long.

Those printer commands which can be used in the HEADER and FOOTER formats are:

MOn,MEn	LEFT MARGIN	(5.9)
MOn,MEn	LEFT MARGIN	(5.9)
Ln	LINE SPACING	(5.10)
Jx	JUSTIFY MODE SET	(5.11)
ILn,IRn	TEXT INDENTING	(5.12)
Qn	CHARACTER PITCH	(5.13)
Dn	CHARACTER DENSITY	(5.14)
GRn	GO RELATIVE	(5.19)
C...,CL...	PRINTER CONTROL CODES	(5.20)

Also, any PROGRAMMABLE CHARACTER (5.21) can be used inside HEADERS and FOOTERS but should be defined outside of the HEADER/FOOTER format.

5.24. BEGIN FOOTER FORMAT FO,FE

See section 5.23 for explanation.

5.25. END HEADER/FOOTER FORMAT Z

This command is used to terminate the HEADER FORMAT and/or FOOTER FORMAT. It should be put on a printer command line by itself. See 5.23 for example.

5.26. PAGE NUMBER SET

This printer command sets the starting page number. It can be changed at any place in the text if desired. The default for the function is "P1".

Example: ("..." means possible other commands)

```
        █...,P100,...■
```

All pages printed after this command will be numbered counting from 100.

Caution: It is usually best to position the PAGE NUMBER SET command at a point in the text where it will be processed someplace between the first line and the page number line of the page for which the number will apply. The only way to insure this is to put the PAGE NUMBER SET command in a printer command line ahead of the first text line to set the page number initially, and on the very next line following a TOP-OF-FORM command (5.18)

Example: ("..." means possible other commands)

```
        █...,G1,...■
        █...,P281,...■
```

The "G1" command will cause the current page to finish printing and the following page will start numbering at 281.

5.27. PAGE NUMBER CHARACTER

You will notice the "#" character was used in the HEADER FORMAT example (5.23) to locate the place were LeScript will automatically put the page numbers for you. If for any reason you would rather LeScript use a different character for that purpose, you can change it with this printer command. The "x" character can not be a space, return character, comma, or a digit (0-9).

Example: ("..." means possible other commands)

```
▮...,P$,...▪
```

In this example, LeScript would assume that any place a group of dollar signs ($) is found in the HEADER or FOOTER you wish the page number to be written. Always use enough of the characters in a group to accommodate the maximum number of digits your largest page number is going to have.

5.28. FORM LETTER DATA FILE SPECIFICATION DATAFILE=filename

Use this printer command at the very top of a FORM LETTER TEXT FILE to tell LeScript the file name of the FORM LETTER DATA FILE containing the data records that will be used as input to the FORM LETTER TEXT FILE at the time of the form letter printing. See section 6 for a complete explanation of how to create and run FORM LETTER TEXT FILES and FORM LETTER DATA FILES.

The "DATAFILE=" in this expression is the constant, but can, however, be shortened to "DA=" if you wish. The "filename" in this expression is the real name that you have given to the FORM LETTER DATA FILE to be used with this FORM LETTER TEXT FILE.

If this printer command is present, then, at the time of the PRINTING (4.34) or VIEWING (4.41) of the FORM LETTER TEXT FILE, LeScript will load the FORM LETTER DATA FILE into memory one data record at a time and replace the data labels within the FORM LETTER TEXT FILE with the data fields from each data record of the FORM LETTER DATA FILE. Once all the replacing is finished for the first data record, the text is printed, and the process is repeated, giving one printed letter (mailing label, document, lease, whatever) for each data record of the FORM LETTER DATA FILE. If this printer command is not present, LeScript assumes that the text file is not a FORM LETTER TEXT FILE and, at the time of PRINTING or VIEWING, will print out the text file only once and without any replacement of the data labels.

Example: ("..." means possible other commands)

```
    ■...,DATAFILE=FORM/DAT,...■
```

This printer command tells LeScript that the text file in memory is a FORM LETTER TEXT FILE, containing data labels that will be replaced by data fields, which are contained in the data records found on the disk under the file name of "FORM/DAT".

5.29. FORM LETTER LABEL NAMES .. LABELS=x;y;z;....

This command is necessary if you are printing FORM LETTERS. It should be used in your FORM LETTER TEXT FILE ahead of the text. This command is how you tell LeScript what the "labels" are within your text that will later get replaced with the data from your FORM LETTER DATA FILE. Every occurrence within your FORM LETTER TEXT FILE of the first LABEL in this command will get replaced with the first data field of each of the data records within your FORM LETTER DATA FILE, the second LABEL for the second data field, and so on. The number of data fields within the data records of your FORM LETTER DATA FILE should be greater than or equal to the number of LABELS that are defined in this command in your FORM LETTER TEXT FILE.

The "x", "y", "z" in this expression are the LABELS as you have defined them for this command. They can be anything you like them to be, 1 to 20 characters long, no spaces, no commas, no semicolons, and no return characters.

Example: ("..." means possible other commands)

```
        ■...,LABELS=?a;?b;?c,...■
```

The command in this example tells LeScript you wish every occurrence of "?a" (no quotes) in your FORM LETTER TEXT FILE to be replaced with the first data field of the records in your FORM LETTER DATA FILE. The second data field replaces each occurrence of "?b". The third data field replaces each occurrence of "?c".

5.30. FORM LETTER DATA ID QUALIFIER D=abc;xyz;@*%....

This is an optional printer command used within a FORM LETTER TEXT FILE to specify which data records from the FORM LETTER DATA FILE will be used as data inputs during the form letter printing run. Only those data records that have ID's matching one of those specified in this printer command and data records which have no ID's at all will be printed. If this command is omitted from the FORM LETTER TEXT FILE, there will be one copy of the FORM LETTER TEXT FILE printed for every data record of the FORM LETTER DATA FILE, regardless of the ID assigned to the data records.

This same printer command is also used in the FORM LETTER DATA FILE just ahead of each data record you wish to give a qualifying identity to. Any record can have one or more ID's.

The "ID=" is the constant in this expression. The "abc", "xyz", "@*%" are data record ID's (separated by semicolons), which are to match ID's of the data records contained within the FORM LETTER DATA FILE, for records that you want to be used as input to the FORM LETTER TEXT FILE at the time of printing. A record ID can be 1 to 20 characters long. There is no limit to the number of data record ID qualifiers that can be used in a FORM LETTER TEXT FILE as long as they all can fit in one printer command line. The data record ID can have any character except a space, comma, semicolon or return character. If used, this command must be positioned in the FORM LETTER TEXT FILE ahead of any printed text. See section 6 for a complete explanation of how to create and run FORM LETTER TEXT FILES and FORM LETTER DATA FILES.

Example: ("..." means possible other commands)

```
    ▮...,DATAFILE=FORM/DAT,ID=WESTCOAST,MIDWEST,...▪
```

This printer command tells LeScript that the text file in memory is a FORM LETTER TEXT FILE, containing data labels that will be replaced by data fields, which are contained in data records, which are found on the disk under the file name of "FORM/DAT". But, only those records that have an ID of "WESTCOAST", "MIDWEST", or no ID at all will be printed. The other data records contained within FORM/DAT will be skipped (not printed).

6. PROCESSING FORM LETTERS

A FORM LETTER, as it is spoken of here, is any text of which several copies have to be printed. Each copy will be the same as all the other copies of the run except for a couple of words or phrases here and there. The "variables" in the text might be such things as persons' names, addresses, occupations, etc. This happens most frequently with form letters, but applies just as well to legal documents, rental leases, invitations, mailing labels, and the like.

LeScript has the capability of processing form letters through the implementation of two special file types: the FORM LETTER TEXT FILE and the FORM LETTER DATA FILE. These two files are described on the following pages.

To quicken your understanding of how LeScript processes form letters, we have included on the LeScript master disk a sample FORM LETTER TEXT FILE called "FORM/TXT" and a FORM LETTER DATA FILE called "FORM/DAT" to be run with it. We recommend that you look them both over, examine the format of each, and then run it for yourself on your own system. This can be done simply by loading "FORM/TXT" into memory using CLEAR-G (4.24), then hit CLEAR-P to start it. You may notice that the data file contains three data records but only two of them were used. Of the two that were used, one had an ID that matched one of the characters in the FORM LETTER DATA ID QUALIFIER printer command (5.30) and the other one had no ID at all. Take out the FORM LETTER DATA ID QUALIFIER printer command in the FORM LETTER TEXT FILE (FORM/TXT), write the file back to disk (CLEAR-F), and run it again (CLEAR-P). This time a copy of the letter for all three data records will be printed.

6.1. CREATING A FORM LETTER TEXT FILE

There are only a few things about this type of text file that are different from your everyday LeScript text file. The main difference is that a FORM LETTER TEXT FILE will contain the FORM LETTER DATA FILE SPECIFICATION printer command (5.28). This command tells LeScript two things: that this is a FORM LETTER TEXT FILE and what the name of the FORM LETTER DATA FILE is that will be used with it. The second difference is that the FORM LETTER TEXT FILE has a FORM LETTER LABEL NAMES printer command (5.29). This tells LeScript what the "variables" are called in your FORM LETTER TEXT FILE that will be replaced at print time with data from the FORM LETTER DATA FILE. The third difference is that the FORM LETTER TEXT FILE may (it's optional) contain the FORM LETTER DATA ID QUALIFIER printer command (5.30) if only a portion of the FORM LETTER DATA FILE is to be used. See 6.4 for an example of how a FORM LETTER TEXT FILE can be created.

6.2. FORM LETTER DATA FILE

The FORM LETTER DATA FILE is the file that contains the list of names, addresses, phone numbers, etc, that will replace the variable labels in the FORM LETTER TEXT FILE. The FORM LETTER DATA FILE is broken into sub-units which we call data records. When the form letter is run, one copy of the FORM LETTER TEXT FILE will be printed for each data record of the FORM LETTER DATA FILE. Therefore, one data record will contain all the necessary data for one person (client, account, whatever). You may, however, choose only a select grouping of data records to be used during the form letter run, if you desire, by giving each data record one or more ID's and then specifying which data records will be used by the FORM LETTER DATA ID QUALIFIER printer command (5.30). For example, all the data records for males can be given the letter "M" or "MALES" or "MEN" for an ID, and all the data records for females can be given the letter "F" or "WOMEN" for an ID. There is no limit to the number of data records you are allowed per FORM LETTER DATA FILE.

Each data record is also made up of sub-units, which we call data fields. The data field is one unit of information about the person (client, account, whatever). There could be one data field for his name, one for his phone number, one for his shoe size, etc. There is no limit as to the number of data fields you are allowed per data record. LeScript allows a data field to be as long or as short as you want. Thus, one label from the form letter text file can be replaced by a paragraph or more.

6.3. CREATING A FORM LETTER DATA FILE

Creating a FORM LETTER DATA FILE is as easy as falling off a log. First, decide whether you are going to type cast the data records allowing yourself the opportunity to specify only records of certain types to be used during the form letter run (5.30). If so, type a printer command character (▮, 4.16), then the characters "ID=" (no quotes), then a 1 to 20 characters ID. If you think the record could belong to two distinct classes or more, type a semicolon (;) and another ID and more semicolons and more ID's as you need them. Do this at the start of every data record in the FORM LETTER DATA FILE. If, on the other hand, you have no need to segregate the data records into group types, then skip this step.

The next step is even easier. Type one unit of data, 1 to 1000 characters long (the guy's name or whatever), then type a RETURN character (▪). Repeat this for each data field of the data record. LeScript assumes the return character to be the end of a field in the data record. If you wish the data record to contain return characters, they must be preceded by END-OF-SENTENCE CHARACTERS (▬, 4.49). However, the data field must still be terminated with one return character that is not preceded by an END-OF-SENTENCE character.

The last step is the easiest of all. Just type one more RETURN character (▪) after the RETURN character of the last data field of the data record. When LeScript sees these two RETURN characters in a row it will know that the data record is ended. Continue these steps, creating one data record for each copy of the form letter to be printed.

The next two pages contain one FORM LETTER TEXT FILE example and one FORM LETTER DATA FILE example. These are the same two form letter files supplied on your LeScript master disk. You will notice that the data file contains three data records each having eight data fields. The first and third data records have ID's, but the second has none.

```
▮jr,datafile=FORM/DAT,ID=MAN■
▮LABELS=?a;?b;?c;?d;?e;?f;?g;?h■
                                        DOWNTOWN FEDERAL TRUS
                                                100 Penny Lan
                                     Fort Knox, Kentucky 4580
▮jb■
■
■
■
■
?b ?c■
?d■
?e,■?f                                        December■12,■19
▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒■
■
■
■
■
■
■
Dear ?a ?c,■
■
■
     We  represent a company that is in the business of loani
money to qualified individuals.■ It has came  to  our  attenti
that  you  are in the market for a new ?g but need ?h to pay f
it.■ We would like to lend you  that  ?h.■  Just  fill  out  t
enclosed application form and send it in today.■ If you qualif
you  could  be  enjoying that new ?g in less than one month fr
today.■ Just think of it, ?a ?c, you could be the  pride  of
with  that  new  ?g.■ Just send in the application, we'll do t
rest.■ You will be notified  of  your  acceptance  by  certifi
mail.■
■
■
■
■
■
■
■
▮il24■
                       Sincerely,■
                       ■
                       ■
                       C. W. Moneybags■
                       President, Downtown Federal Trust■
```

```
▌ID=MOVIE-START;MAN■
Mr.■
Clark■
Kent■
1655 Lois Lane■
Metropolis■
NY▬▬▬10507■
cape■
$50.00■
■
Miss■
Sally■
Somethingelse■
38-24-36 Waahoo Court■
Long Beach■
CA▬▬▬93228■
bikini■
$200■
■
▌ID=BIG-SPENDER■
Mrs.■
Hellen■
Housewife■
250 Taken Place■
Contentsville■
IN▬▬▬59764■
dish washer■
$750■
■
```

7. EDITING BASIC AND EDTASM FILES

Almost every capability available to you while editing normal text files can also be used when editing BASIC and EDTASM files. A few things work a little differently due to the unique nature of these types of files. There are also a couple additional capabilities LeScript reserves just for these two file types. We are sure that as you become familiar with these few special provisions you will soon be using the same high-speed, labor-saving powers of LeScript to edit your program source files as you have for your text files.

7.1. WIDTH FIELD

The contents of the WIDTH FIELD (3.2 and Appendix A) will determine whether LeScript treats your text as normal text or as a BASIC or EDTASM file. If the WIDTH FIELD contains "BAS", the text is treated as a BASIC file. If the WIDTH FIELD contains "EDA", the text is treated as an Apparat EDTASM file. If the WIDTH FIELD contains "EDR", the text is treated as a Radio Shack EDTASM file.

7.2. INSERT PROGRAM LINE

When the OPEN LINE function (CLEAR-O, see 4.33) is implemented while editing a BASIC or EDTASM file, a special thing happens: LeScript goes into the INSERT PROGRAM LINE mode. All the text below the cursor moves down one line, a new line number is automatically written at the start of the new blank line (one greater than the value of the previous line number), and the cursor is moved to the second space after the line number. While in this mode, each time a line is ended by a RETURN character (▪)(striking the ENTER key, 4.47) another blank line with a new line number appears on the screen.

This function will terminate automatically when the line number of the current line is one less than the line number of the following line, or manually by moving the cursor to the start of the line and hitting CLEAR-L (4.30).

7.3. RENUMBERING

Upon hitting CLEAR-8, the BASIC or EDTASM file is removed from the screen, all the program lines are renumbered starting at 10 and counting by 10, and the text returns to the screen. This function is also covered in section 4.60.

7.4. PRINTER COMMANDS

Don't use PRINTER COMMANDS with BASIC or EDTASM files, simply because other programs, such as compilers and assemblers, that read these files, will not allow PRINTER COMMANDS. When you do print the file, LeScript's printer command defaults will take over and give you page lengths of 66 lines, 6 lines of spacing at the top, 8 lines of spacing at the bottom, and 8 characters of spacing at the left.

7.5. LINE LENGTHS

In order to allow those "hardcore" BASIC programmers the ability to type out BASIC program lines as long as their arms, LeScript has preset maximum line lengths to 252 characters for all BASIC files. For editing EDTASM files where line length is not as important, LeScript has preset the maximum line length for these files to the width of your display screen (either 64 or 80, depending on the model or computer).

7.6. FILE TYPE FOR BASIC FILES

LeScript can not read BASIC files that are in compressed format. LeScript can only read BASIC files that are in ASCII format. To make sure your BASIC file is in ASCII format before attempting to load it into LeScript, first do the following:

```
Enter BASIC
Type: LOAD"FILENAME"            (you supply the name)
Type:  SAVE"FILENAME",A
```

Now, that file is on the disk in ASCII format, and LeScript will be able to read it without any problems. Just exit BASIC, start LeScript, and load in the file with CLEAR-G (4.24) as you would any other file. LeScript will know that it is a BASIC file and treat it accordingly.

Note: If you attempt to load a BASIC file into LeScript and "EDR" appears in the WIDTH FIELD and a couple of garbage characters appear in the display portion of the screen, this is an indication that the BASIC file has not yet been converted to ASCII format.

A. EXPLANATION OF FILE TYPES

LeScript can read and write 5 different text file types. They are: LeScript, Apparat EDTASM, Radio Shack EDTASM, BASIC, and ASCII. LeScript can also read, but not write, ZORLOF files and Scripsit files. Each of these file types and the differences that make them unique are explained on the following page.

An ASCII file, as it is spoken of here, is a file, which when stored on disk, has no special codes embedded in it--just viewable text in ASCII format. This is the standard file format of most garden variety word processors.

Because of some of the special functions that LeScript has, it was necessary for LeScript files to go beyond the standard ASCII file format and add status codes at the end of each line within the text files to define that line's justification, width, and indent, so that proper line positioning would be assured as files are read from the disk for viewing.

The ASCII file format feature, present in LeScript, allows the operator the opportunity to save a file onto disk in standard ASCII format instead of the LeScript text file format if so desired. Using this capability, any file created on LeScript can be read by other word processing systems, and vice versa.

To save a file in ASCII format, just type the letters "ASC" in the width field and hit CLEAR-F. To save a file in LeScript text file format, type a number (5-252) in the width field or leave it blank, then hit CLEAR-F. Also, any time a file is retrieved from disk, the width field will contain the file type: ASC--ASCII, BAS--BASIC, EDT--Apparat EDTASM, EDR--Radio Shack EDTASM, or a number (5-252) or blank--LeScript.

LeScript TEXT FILES	The first character is a D5H, followed by three characters that comprise the width field of the top status line, followed by the text, ending with a 00H. Each text line is followed by four line status codes, the first and fourth define its justification, the second its left indent, and the third its width. Identified by LeScript by the leading D5H.
APPARAT EDTASM FILES	The first charater is a D3H, followed by the first 6 characters of the file name, followed by the text, ending with a 1AH. Each line starts with 5 ASCII digits with the 8th bit set (B0H-B9H), words are separated by spaces (20H) and/or tabs (09H), and each line ends with a carriage return (0DH). Identified by LeScript by the leading D3H.
RADIO SHACK EDTASM FILES	This is the same as the Apparat EDTASM file except the D3H and the first 6 characters of the file name are not present in this file type. Identified by LeScript by a leading character of B0H-B9H.
BASIC TEXT FILES	Similar to the Radio Shack EDTASM file type except that the 8th bits of the line number digits are cleared (not set), and the ending character of the file is a 00H. Identified by LeScript by a leading character of 30H-39H.
ASCII TEXT FILES	Similar to the BASIC file type except the first character of the file is any 7-bit ASCII character except a digit (30H-39H). Identified by LeScript by a leading character of 21H-2FH or 3AH-7FH.
ZORLOF TEXT FILES	Same format as the LeScript text files except that the leading character is a D4H, and there are only three status codes following each text line.
Scripsit TEXT FILES	Same format as ASCII except that each paragraph ends with 8EH instead of 0DH.

B. CUSTOMIZING LeScript

Reserved for future releases of LeScript.

C. ANSWERS TO COMMONLY ASKED QUESTIONS

GETTING STARTED

1Q. Can LeScript work on a one disk drive system?
1A. Yes, but with most DOS's, you need two disk drives to copy the LeScript files from the master disk to your operating system.

2Q. How can I copy LeScript to my disk system when I have only one disk drive?
2A. You can either: 1) find a friend who has two disk drives and use his system to do the copying, or 2) ask your local Radio Shack dealer if you could borrow his system for a few minutes to do the copying, or 3) send the master LeScript disk and a backup copy of your operating system to us along with $10.00 for handling, and we will copy all the LeScript files onto your operating system disk.

3Q. Will LeScript work on a Model 1 without the lower case modification?
3A. Yes, but all lower case characters and many other symbols will appear incorrectly on your display screen.

4Q. Can I use my Model 1 disk on a Model 3?
4A. Yes, by using the "CONVERT" utility on the Model 3 TRSDOS to copy the LeScript files over to a Model 3 formatted disk.

5Q. Can I use my Model 3 disk on a Model 1?
5A. Yes, if you have a DOS that is able to perform the conversion process. An example is the "PDRIVE" function on NEWDOS/80, which can be used to copy the LeScript files to a Model 1 formatted disk (consult the NEWDOS/80 manual for the details). Or you can send us your Model 3 disk along with $10.00 handling fee, and we will reprogram it to the most current revision in Model 1 format for you.

6Q. How do I get a file without going to the directory?
6A. Type the file name and hit CLEAR-G to load the file.

EDITING

7Q. Why do functions like PRINT, for example, not work when I type P, -, CLEAR?
7A. To get LeScript to execute functions, you are to hold down the CLEAR key and then hit the other key. In this case the other key would be the P key. Do not hit the dash key.

8Q. How do I force a page to end?
8A. You can force a page break by using the "G1" command which is explained on section 5.18 of the LeScript manual.

9Q. How can I center using expanded width characters in the same text file with normal width characters?
9A. The only way to center expanded width characters along with normal width characters is to make the wide text using CLEAR-X and not the "Qn" command.

PRINTING

10Q. Why do I get line feeds when I want bolded words and underlines?
10A. Either you have not used the right printer specification code (see section 5.1) or you need to disable the automatic line-feed in your printer. If you disable the automatic line-feed, you must use the "LF" command (see section 5.2).

11Q. Underline space (CLEAR-4) does not work with K5 or K20 printer drivers. Why is this?
11A. For some reason Epson printers are internally programmed to ignore the underline command sent from the computer to underline a group of spaces unless there is an underlined non-space character preceding and following that group. We suggest you use the underline character (SHIFT-CLEAR-=) instead.

12Q. When I use "K33" on my Epson or Gemini printer to get proportional-space printing it takes much longer than printing in mono-space mode. Is this normal?
12A. Yes, this is normal. These two printer types were not designed to produce proportional-space printing directly. LeScript gets around that limitation by setting the printer to graphic mode to print the variable width spaces and returns to charcter mode to print the characters. But, due to internal limitations of these two printers, this process takes longer than normal mono-space printing.

13Q. How can I get my printer to print darker?
13A. On printers that have "emphasized" printing, it is possible to get an entire letter to print darker by using the "CL" command on LeScript (see section 5.20).

14Q. Can LeScript's Form Letter function work with data files containing names and addresses generated with some other program?
14A. Yes, provided the format of the other data file is the same as that for required for LeScript. This format is shown on section 6.5 of this manual.

15Q. Can LeScript's Form Letter Function be used to print out mailing labels from a data file containing lists of names and address.
15A. Yes, the same way it prints out form letters from a list of names and addresses. Just write a text file that has a printer command line at the top that correctly defines top line, end line and depth of one label with the "Tn", "En", and "Yn" commands respectively, then put the field labels under that, and run it.

FILES

16Q. How are files in BASIC handled with LeScript?
16A. Files written in BASIC must have been first saved to disk in ASCII format (as opposed to compressed format) in order for LeScript to read them.

17Q. When I try to load a BASIC file into LeScript all I get is "EDR" in the width field and a few odd characters in the text portion of the display. Why is that?
17A. The BASIC file is in compressed format and must first be saved in ASCII format in order for LeScript to be able to read it.

SPELLING CHECKERS

18Q. What spelling checkers does LeScript work with?
18A. LeScript is compatible with just about any spelling checker. The files will need to be saved in ASCII format to use the spellng checkers. Type "ASC" in the width field, hit CLEAR-F, reboot the computer, begin the spelling checker program, use it to load and check the file just saved out of LeScript.

CUSTOM PATCHES

19Q. What if I want to have LeScript perform in some non-standard way that is unique to my application? Is it possible to pay to have custom patches like this put into LeScript for me?
19A. Yes. We can and do develop patches special for individual applications. All requests for this service must be submitted in writing in full detail, and stating specifically that you want an "estimate". We will return to you in writing an estimate of the cost.

LeScript EDITING FUNCTIONS

Automatic Search & Replace (also can be automatic search & delete if RP field blank)	CLEAR - A	p. 2
Blank Character (▮ , used as an inflexible space)	CLEAR - SPACE	p. 3
Block Marker Character (▀ , sets beginning & end of block)	CLEAR - B	p. 2
Bold	CLEAR - 3	p. 34
Capital Lock (on/off)	CLEAR - /	p. 2
Clear (resets everything)	CLEAR - 0	p. 3
Copy Block (first define block with CLEAR-B, the copy is to cursor location)	CLEAR - C	p. 2
Cursor Down	↓	p. 3
Cursor Far Left	SHIFT - ←	p. 2
Cursor Far Right	SHIFT - →	p. 2
Cursor Left	←	p. 2
Cursor Right	→	p. 2
Cursor Up	↑	p. 2
Delete Block (from cursor to block marker character)	CLEAR - *	p. 3
Delete Character Backwards	CLEAR - ←	p. 2
Delete Character Forwards	CLEAR - →	p. 2
Delete Line (also can be used to remove entries from print queue)	CLEAR - L	p. 2
Delete Word	CLEAR - W	p. 3
Directory (hit number key for desired drive, space bar to page, CLEAR-Z to exit)	CLEAR - D	p. 2
End-Of-Sentence Character (▬ , used as an inflexible space between sentences)	CLEAR - ENTER	p. 3
End of Text	CLEAR - E	p. 2
Exit To DOS READY	SHIFT - CLEAR - 0	p. 3
Expanded Characters Mode (on/off)	CLEAR - X	p. 3
File Text To Disk (filespec must be in Name Field)	CLEAR - F	p. 2
Get Text From Disk (filespec must be flushed left on cursored line, directory OK)	CLEAR - G	p. 2
Home Cursor (top left corner of screen)	CLEAR - H	p. 2
Insert Mode (on/off)	CLEAR - I	p. 2
Italic	CLEAR - 5	p. 3
Keyboard Click	CLEAR - 9	p. 3
Kill File On Disk (position cursor on filespec within directory, double acting)	CLEAR - K	p. 2
Lower Case	CLEAR - .	p. 22
Move Block (first define block with CLEAR-B, the move is to cursor location)	CLEAR - M	p. 27
Open Line (creates a blank line at cursor)	CLEAR - O	p. 2
Page Down	CLEAR - ↓	p. 2
Page Up	CLEAR - ↑	p. 2
Pause Character (printer will pause, hit spacebar to resume)	CLEAR - 6	p. 3
Printer Command Character (▮ , used to begin printer command lines)	CLEAR - ;	p. 2
Print Screen (text in display format goes to printer)	CLEAR - @	p. 24
Print Text (text goes to line printer in final form)	CLEAR - P	p. 2
Queue Files For Chained Printing (position cursor on file within directory)	CLEAR - Q	p. 27
Rejustify Text	CLEAR - J	p. 2
Renumber (for EDTASM and BASIC files, starts at 10, counts by 10)	CLEAR - 8	p. 3
Replace (replace string must be typed in RP field)	CLEAR - R	p. 2
Remove Block Markers	CLEAR - Q	p. 2
Return Character (▪ , used to terminate lines)	ENTER	p. 3
Scroll Down	SHIFT - ↓	p. 2
Scroll Up	SHIFT - ↑	p. 2
Search (search string must be typed in SH field, blank characters are wild)	CLEAR - S	p. 2
Split Text (left of cursor scrolls up, right of cursor scrolls down)	CLEAR - Y	p. 3
Status Line Home (moves cursor to Name Field)	CLEAR - =	p. 3
Store Block to Disk (first define block with CLEAR-B, put filespec on blank line)	CLEAR - N	p. 2
Subscript	CLEAR - 2	p. 34
Superscript	CLEAR - 1	p. 34
Tab (interpretted as kill file function if done in directory)	CLEAR - K	p. 2
Terminate (exits from CLEAR-@, -A, -D, -P, -S, -V)	CLEAR - Z	p. 3
Top of Text	CLEAR - T	p. 3
Underline	CLEAR - 4	p. 3
Upper Case	CLEAR - ,	p. 2
View Text (look at text in near final form without using printer)	CLEAR - V	p. 3

Page85

LeScript PRINTER COMMANDS

Automatic Line-feed Insertion (CR+LF when used, default CR only)	LF	p. 44
Baud Rate (n/1=110, 2=150, 3=300, 4=600, 5=1200, 6=2400, 7=4800, 8=9600, set for serial when used, default parallel)	Rn	p. 45
Character Density (n=# of characters/inch for proportional-space printing, set from 6.0 to 20.0 in steps of .1, default 13.0 dot matrix/10.0 daisy, must follow "K" setting)	Dn	p. 54
Character Pitch (n/1=17,2=15,3=12,4=10,5=8.5,6=7.5,7=6,8=5)	Qn	p. 53
End Text Line Set (n=end line position, default 62)	En	p. 48
Form Letter Data File Specification (used in text file to specify name of data file)	DATAFILE=filename	p. 68
Form Letter Record ID Qualifier (used in text file to select which data records used)	ID=abc;xyz;...	p. 70
Form Letter Label Names (used in text file to assign variable label names)	LABELS=x;y;z;...	p. 69
Footer Format Begin (same as header format begin)	FO,FE	p. 65
Go Relative (Causes n number of lines to be skipped before printing continues)	GRn	p. 58
Go To Line Number (printing continues at line number specified by n)	Gn	p. 57
Header Format Begin (starts the header format printer commands)	HO,HE	p. 63
Header/Footer Format End (last printer command in the header/footer format)	Z	p. 65
Indenting (n=# of character widths of indenting, default IL0,IR0)	ILn,IRn	p. 52
Individual Sheet Pausing (causes printer to stop at end each sheet, printer's on-line switch or space bar on keyboard to continue)	I	p. 46
Justify Mode Set (x/L=left,R=right,B=both,C=center, default L)	Jx	p. 51
Legal Courtroom Line Numbering (turns on auto-line-numbering)	N	p. 62
Line Spacing (n=spacing, example 2=double spacing, default 1)	Ln	p. 50
Margin (O=odd,E=even, n=left margin in character widths, default both 8)	MOn,MEn	p. 49
Off/On Printing (text between OFF and ON will be ignored during printing)	OFF,ON	p. 56
Page Number Character (x=character to use instead of "#")	Px	p. 67
Page Number Set (n=starting page number, can be changed at any time)	Pn	p. 66
Printer Control Codes (aabb...=Hex pair printer codes, n=# of times to be sent, "L" is optional use if sent before every line, all type fonts of Epson with this command)	CLaabb...Xn	p. 59
Printer Type (see list below)	Kn	p. 40
Sheet Size Set (n=length of paper in # of lines, default 66)	Yn	p. 47
Suppress Line-Feed (causes one line to overstrike another)	XL	p. 55
Tabs Set(a-f...=tab stops, use only once per file)	TABa;b;c;d;e;f...	p. 46
Top-Of-Form	G1	p. 57
Top Text Line Set (n=top line position, default 7)	Tn	p. 47

PRINTER LIST

p=Proportional-space m=Mono-space

K0	m	Any plain jane printer
K1	m	DWII, DWP-210, DWP-410
K2	p	DWII
K3	p	DWP-210, DWP-410
K4	m	C.Itoh FP-1500, F-10, HR-15, Diablo 630, 1610, JUKI 6100, BMC-501, QUME, NEC 3515, 3525, 5515, 5515, 7715, 7725, DP-55, EXP-500, EXP-550.
K5	p	C.Itoh FP-1500, F-10, HR-15, Diablo 630, 1610, JUKI 6100, BMC-501, QUME, NEC 3515, 3525, 5515, 5515, 7715, 7725, DP-55, EXP-500, EXP-550.
K6	m	NEC 3510, 3520, 5510, 5520, 7710, 7720
K7	p	NEC 3510, 3520, 5510, 5520, 7710, 7720
K10	m	Daisy Writer 1500, 2000
K11	p	Daisy Writer 1500, 2000
K12	m	DTC 380Z
K13	p	DTC 380Z
K14	m	HR-1, CR-1
K16	m	TP-1
K30	m	MX-80
K32	m	MX-80/Gt+, MX-100, RX-80, FX-80, FX-100, Gemini 10, 10x, 15
K33	p	MX-80/Gt+, MX-100
K35	p	RX-80, FX-10, FX-100
K37	p	Gemini 10, 10x, 15
K40	m	Microline 80, 82, 83
K42	m	Microline 84, 92
K43	p	Microline 84, 92
K50	m	Prowriter 8510, 8515, NEC 8023, PMC-8510, TEC 8500R, DMP-85
K51	p	Prwriter 8510, 8515, PMC-8510
K53	p	NEC 8023
K55	p	TEC 8500R, DMP85
K60	m	LPIV, 737, 739
K61	p	LPIV, 737, 739
K62	m	LPVI
K64	m	LPVIII
K65	p	LPVIII
K66	m	DMP-100
K68	m	DMP-200, -400, -500
K69	p	DMP-200, -400, -500
K70	m	DMP-2100
K71	p	DMP-2100
K80	m	Toshiba P1350
K81	p	Toshiba P1350
K86	m	IDS 480
K87	p	IDS 480
K94	m	BASE-2
K96	m	MPI 88, 99, 150

LeScript ADDENDUM

New features not mentioned elsewhere in the manual

1. KSM EMULATION

LeScript has the ability do KSM (Key Stroke Multiply) emulation similar to LDOS. You hit a key and a whole phrase is inserted at the place your cursor is. You can program up to 55 different phrases of any length into LeScript for immediate access anytime you want them. The keys that can be reprogrammed are:

SHIFT-CLEAR-A thru SHIFT-CLEAR-Z,
SHIFT-CLEAR-1 thru SHIFT-CLEAR-5,
! " # $ % & ' ( ) * + - / < = > ? @ [\ ] ` { }

To define the keys you must "build" a KSM file. Start with a clear screen, type LESCRIPT/KSM in the name field, and type ASC in the width field. Move the cursor to the first text line of the display and type a phrase, end it with a return character (hitting the ENTER key). You have just defined the phrase for SHIFT-CLEAR-A. Type in another phrase under the first one, end it with a return character, and you have defined the phrase for SHIFT-CLEAR-B. The order that they are defined is the same as the order given in the above list. If you wish to skip any key assignments, just put a return character and no phrase in the KSM build file for that key. Any undefined keys will output their normal character when Lescript is running. Also, use ";" characters in phrases any place you wish there to be a line ender, as these will get translated to a return characters at run time.

Now that the SM file is built, hit CLEAR-F to file it to disk, hit SHIFT-CLEAR-0 twice to leave LeScript, then type LESCRIPT and hit the ENTER key. The DOS will load in LeScript, then LeScript will take over and load in the KSM file (LESCRIPT/KSM), then you will be up and running. Now if you hit SHIFT-CLEAR-A, the entire first phrase of your KSM file will type itself onto the screen. SHIFT-CLEAR-B will give you the second phrase, and so on.

For your convenience we have supplied a sample KSM file on the LeScript master disk. On the master disk it is called SAMPLE/KSM. IF you rename this file to LESCRIPT/KSM and run LeScript you can see how this powerful feature works without having to create a KSM file yourself. This sample file programs phrases into SHIFT-CLEAR-A thru SHIFT-CLEAR-Z, the other reprogrammable keys will function as they were.

Page 87

It is also possible to combine all your commonly used reprogrammable characters (see section 5.21) in this same KSM build file. The rule that LeScript uses to tell the difference between a reprogrammable key definition and a reprogrammable character definition is: If the definition for a given key/character assignment in the KSM build file starts with the same character as would normally be produced by that key then the definition is for reprogramming the <u>character</u> (same as feature described in section 5.21), otherwise it is for reprogramming the <u>key</u> (KSM emulation).

EXAMPLE:

If in the 36th entry of the KSM build file you have:

```
Mary had a little lamb•
```

then this would reprogram the "%" key so that anytime you hit the "%" key LeScript would type "Mary had a little lamb". But if in the 36th entry you had:

```
%1b45;•
```

then this would reprogram the "%" character so that when you hit the "%" key you would get just the "%" character. But when LeScript goes to print that character, the values X'1B' and X'45' would get sent to the printer instead. Note, the ";" character is required to follow any build file entry for reprogramming characters.

2. USE OF THE VERTICAL BAR CHARACTER

The vertical bar character can be used as a regular character in your text if it is followed by a space, tab, blank character, end-of-sentence character, return character, or another vertical bar character. At any other time it is used to expand the character that follows it (see section 4.43). This character is directly enterable from the MAX-80 keyboard. On any other system it can be entered by hitting CLEAR-X, then any character, then CLEAR-X again, then deleting the other character.

3. SPECIAL ENTRY PARAMETERS

Upon entry to LeScript, the user can specify certain operating conditions. To do this type LESCRIPT followed by the parameter character(s).

EXAMPLE:

```
LESCRIPT -
```

This will cause LeScript to use just the lower 64k addressable bytes of memory an leave the upper banks free for other uses. This is especially useful for MAX-80 owners using MDISK. Under TRSDOS 6, if the MEMDISK is installed upon entering LeScript, the top 64K will automatically become unusable by LeScript, without having to type the "-" character.

EXAMPLE:

```
LESCRIPT @
```

This will cause LeScript to use the DOS's routing for the printer instead of its own printer drivers. This is most useful for those who wish to take advantage of the printer spooling capabilities of LDOS and TRSDOS 6.. A word of caution though, most DOS printer drivers "trash" certain byte values that are necessary for the control of your printer running under LeScript. Use this feature at your own risk. Our strongest recommendation is, unless you know that there are no byte values thrown out by your DOS's printer driver, that you not use this feature at all.

EXAMPLE:

```
LESCRIPT -,@
LESCRIPT @,-
LESCRIPT -@
LESCRIPT @-
```

Any one of these will perform both of the above.

EXAMPLE:

```
LESCRIPT *
```

This is already mentioned in the manual but it is appropriate to mention it here as well. This will cause LeScript to enter leaving all your text as it was before you last left LeScript.

4. ENHANCED DIRECTORY FEATURES

When accessing a disk drive's directory under LeScript, the one displayed first will be the same one that was displayed last, instead of drive 0 always being displayed first. This is especially helpful for those who keep all there data files on drive 1 and need not ever see the directory for drive 0.

Typing a file extension at the beginning of the cursored line prior to hitting CLEAR-D will give you a directory listing of just the files on the drive with that extension. LeScript will automatically remove the extension from the text file for you when it calls in the directory. Also, once such a specification has been entered, LeScript will remember it so you need not enter it again until you wish to see a different selection of files from the disk.

EXAMPLE:

```
          /TXT               <CLEAR-D>
```

This will display a directory of only those files that have the "/TXT" extension.

```
          /                  <CLEAR-D>
```

This will display a directory of only those files that have no extension.

```
          /*:1               <CLEAR-D>
```

This will display a directory of all files on the disk in dive 1.

5. HEADERS, FOOTERS AND MARGINS

IF you wish to have the exact same definition for both the even-numbered page header and the odd-numbered page header you need only use the printer command "H" followed by the definition, instead of "HE" and "HO" (see section 5.23). For footers, "F" followed by a definition will define both footers at the same time (see 5.24). Use the printer command "Mn" to define both margins at the same time (see 5.9).

6. PROFILE DATAFILE TRANSLATOR PROGRAM

The LeScript master disk now contains a public domain Proofile datafile translator program. This program was writen by Scott Loomer of MicroConsultans, 315 Palomino Lane, Madison, Wisconsin (608-233-7739).

To run the program just enter BASIC and type:

```
LESTRIP <ENTER>
```

The programm will ask you to enter the name of the Profile formated datafile (this is the input), and then the name for the LeScript formated datafile (this is the output). Then the program takes over and does the translation. Now your Profile datafiles containing your address lists can be used with LeScript's Form Letter feature.

This is a public domain program, and as such is not supported by ANITEK Software Products. Any questions or suggestion concerning this program should be directed to Scott Loomer at the above address.

7. ADDITIONAL PRINTER DRIVERS

K20	NEC 7710 and 7720 (mono-space mode)
K21	NEC 7710 and 7720 (proportional-space mode)
K34	RX-80, FX-80, and FX-100 (mono-space mode)
K36	Gemini 10, 10X, and 15 (mono-space mode)

8. ADDITIONAL KEY ASSINGMENTS

SHIFT-ENTER	= End-Of-Sentence character
SHIFT-0	= Cap-Lock (toggles)
SHIFT-CLEAR-↓	= Disable Blink (toggles)

9. UPPER CASE LESCRIPT

The LeScript master disk is now being supplied with an upper case only version of LESCRIPT, called LESCRIPT/UC. This is entended for users of TRS-80 Model I computers without the lower case modification installed. This is exactly the same program as LESCRIPT/CMD except that when entering characters, they will all be in upper case. To run this version of the program just type (at DOS READY): LESCRIPT/UC.

CLEAR - =	MOVE CURSOR TO NAME FIELD
CLEAR - Ø	CLEAR FILE
SHIFT-CLEAR-Ø	EXIT TO DOS
CLEAR-W	DELETE WORD
CLEAR-→	DELETE CHARACTER
CLEAR-/	CAPITAL LOCK
CLEAR-E	PUT CURSOR AT END OF TEXT
CLEAR-H	HOME CURSOR
CLEAR-I	INSERT TEXT AT CURSOR
CLEAR -F	SAVE FILE ON DISK
CLEAR -G	LOAD FILE FROM DISK
CLEAR -P	PRINT TEXT
CLEAR -B	INSERT BLOCK MARKER
CLEAR -D	REMOVE BLOCK MARKERS
CLEAR -C	~~MOVE~~ COPY BLOCK
CLEAR -M	MOVE BLOCK
CLEAR -S	SEARCH FOR STRING IN SEARCH FIELD
CLEAR -R	REPLACE STRING FOR STRING IN REPLACE FIELD

CLEAR + -
CLEAR + / Capital Letters only

[illegible]
CLR+K = Tab
TAB 9;18;27;36;45
CLR+T
CLR → Delete character
CLR+@ Print screen
CLR+F Save Text
CLR+G Load Text
CLR+P [illegible]

Start using Lescript
1. Press CLEAR + / if you want capital letters only else flip Lower case switch up

Print file in memory
1. Press CLEAR + P

Save file to disk
1. Press CLEAR + -
2. Type file name you want
3. Type CLEAR + H
4. Type CLEAR + F